**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

NGUYỄN THANH HỮU MSSV: N20DCCN108 Tên Đề tài: Ứng dụng dự đoán khách hàng rời bỏ dịch vụ bảo hiểm

Lớp:D20CQCNPM02-N

**Gáy**

Mẫu trang bìa

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

------------------------------



**BÁO CÁO THỰC TẬP**

**TỐT NGHIỆP ĐẠI HỌC**

***Đề tài*:“ ỨNG DỤNG DỰ ĐOÁN XU HƯỚNG KHÁCH HÀNG RỜI BỎ DỊCH VỤ BẢO HIỂM VỚI MÔ HÌNH SUPER LEARNING”**

**Người hướng dẫn : TS. HUỲNH TRỌNG THƯA**

**Sinh viên thực hiện : NGUYỄN THANH HỮU**

**Mã số sinh viên : N20DCCN108**

**Lớp : D20CQCNPM02-N**

**Khoá** **: 2020-2024**

**Ngành : KỸ THUẬT PHẦN MỀM**

**Hệ** **: ĐẠI HỌC CHÍNH QUY**

**TP.HCM, tháng 08/2024**

TP. HCM

2024

**BỘ THÔNG TIN VÀ TRUYỀN THÔNG**

**HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG**

-----------------------------



**BÁO CÁO THỰC TẬP**

**TỐT NGHIỆP ĐẠI HỌC**

***Đề tài*: “ ỨNG DỤNG MÔ HÌNH SUPER LEARNING ĐỂ DỰ ĐOÁN XU HƯỚNG KHÁCH HÀNG RỜI BỎ DỊCH VỤ BẢO HIỂM”**

**Người hướng dẫn : TS. HUỲNH TRỌNG THƯA**

**Sinh viên thực hiện : NGUYỄN THANH HỮU**

**Mã số sinh viên : N20DCCN108**

**Lớp : D20CQCNPM02-N**

**Khoá** **: 2020-2024**

**Ngành : KỸ THUẬT PHẦN MỀM**

**Hệ** **: ĐẠI HỌC CHÍNH QUY**

**TP.HCM, tháng 08/2024**

**LỜI CẢM ƠN**

Lời đầu tiên, em xin gửi lời tri ân sâu sắc đến các thầy cô trường Học Viện Công Nghệ Bưu Chính Viễn thông cơ sở tại TP.HCM đã tận tình dẫn dắt và truyền đạt cho em rất nhiều kiến thức quý báu trong các học kỳ vừa qua.

Đặc biệt, em xin gửi lời cảm ơn chân thành đến TS. Huỳnh Trọng Thưa. Thầy đã hướng dẫn tận tình, truyền đạt kiến thức, chỉ bảo cho em trong suốt thời gian học tập và thực hiện đề tài. Kính chúc thầy và gia đình nhiều sức khoẻ và thành công trong cuộc sống.

Và xin chân thành cảm ơn tất cả các bạn đã luôn sát cánh, giúp đỡ, động viên mình

trong những thời điểm khó khăn, tiếp thêm động lực và ý chí giúp vượt qua khó khăn trong suốt quá trình học tập tại trường.

Tuy nhiên, vì thời gian và kiến thức chuyên môn còn hạn chế cùng nhiều yếu tố khách quan khác nên trong quá trình thực hiện đề tài không tránh khỏi những sai sót. Kính mong được lời nhận xét và góp ý của thầy cô, để em có thể hoàn thiện bản thân mình hơn.

Một lần nữa, em xin chân thành cảm ơn!

TP.HCM, tháng 07 năm 2023

Sinh viên thực hiện

Nguyễn Thanh Hữu

**MỤC LỤC**

**DANH MỤC CÁC BẢNG VÀ SƠ ĐỒ HÌNH**

**KÍ HIỆU CÁC CỤM Ừ VIẾT TẮT**

**MỞ ĐẦU**

Trong bối cảnh thị trường ngày càng cạnh tranh, việc duy trì và gia tăng lượng khách hàng trung thành trở thành một yếu tố thì chốt quyết định sự thành công của doanh nghiệp. Tuy nhiên, không phải lúc nào doanh nghiệp cũng có thể giữ chân được tất cả khách hàng, và việc dự kiến khách hàng có nguy cơ rời bỏ (churn) là một công thức không nhỏ. Sự phát triển cấp cao của công nghệ trí tuệ nhân tạo (AI) đã mở ra những cơ hội mới để giải quyết vấn đề này một cách hiệu quả hơn.

Mô hình AI dự đoán khách hàng sẽ đi là một công cụ mạnh mẽ giúp doanh nghiệp nhận diện sớm các dấu hiệu của khách hàng có thể bỏ đi. Bằng cách phân tích lịch sử và hành động dữ liệu của khách hàng, AI có khả năng đưa ra những thông tin chính xác được mong đợi về nguy cơ rời đi của từng khách hàng. Từ đó, doanh nghiệp có thể phát triển các biện pháp thích hợp để giữ chân khách hàng, cải thiện trải nghiệm của họ và nâng cao giá trị vòng đời khách hàng.

Việc ứng dụng mô hình AI vào dự đoán khách hàng đi không chỉ giúp doanh nghiệp tối ưu hóa chiến lược kinh doanh mà tiết kiệm được chi phí tiếp thị và chăm sóc khách hàng. Điều đặc biệt quan trọng trong bối cảnh nguồn lực hạn chế và yêu cầu hoạt động hiệu suất tối ưu.

Vì vậy, trong đề tài này tôi sẽ triển khai một ứng dụng đơn giản với mô hình super learning để đưa ra các dự đoán chính xác nhất cho hệ thống dự đoán khách hàng rời bỏ dịch vụ bảo hiểm.

**CHƯƠNG 1: GIỚI THIỆU ĐỀ TÀI**

**1.1 Tổng quan về đề tài**

### **1.1.1** Lý do chọn đề tài.

Trong thời đại số hóa hiện nay, việc giữ chân khách hàng trở thành một thách thức lớn đối với các công ty bảo hiểm. Khách hàng rời bỏ dịch vụ không chỉ ảnh hưởng đến doanh thu mà còn gây tổn hại đến hình ảnh và uy tín của công ty. Việc hiểu và dự đoán xu hướng khách hàng rời bỏ dịch vụ là rất quan trọng, giúp các công ty bảo hiểm có thể đưa ra các biện pháp phòng ngừa và cải thiện chất lượng dịch vụ.

Một trong những phương pháp tiên tiến để dự đoán xu hướng này là sử dụng mô hình Super Learning, một dạng của phương pháp ensemble learning. Super Learning kết hợp nhiều mô hình khác nhau nhằm tạo ra một mô hình dự đoán mạnh mẽ và chính xác hơn. Việc áp dụng mô hình Super Learning trong dự đoán xu hướng khách hàng rời bỏ dịch vụ bảo hiểm có thể mang lại nhiều lợi ích thiết thực, bao gồm:

1. **Cải thiện độ chính xác của dự đoán**: Mô hình Super Learning kết hợp sức mạnh của nhiều mô hình đơn lẻ, từ đó tăng cường khả năng dự đoán và giảm thiểu sai số.
2. **Đưa ra các biện pháp phòng ngừa hiệu quả**: Với các dự đoán chính xác, công ty bảo hiểm có thể phát hiện sớm các khách hàng có nguy cơ rời bỏ dịch vụ và đưa ra các biện pháp chăm sóc, khuyến mãi, hoặc cải thiện chất lượng dịch vụ để giữ chân khách hàng.
3. **Tối ưu hóa chiến lược kinh doanh**: Thông tin từ dự đoán xu hướng rời bỏ dịch vụ có thể giúp công ty điều chỉnh chiến lược kinh doanh, từ đó tăng cường khả năng cạnh tranh và nâng cao hiệu quả hoạt động.
4. **Tiết kiệm chi phí**: Việc giữ chân khách hàng hiện tại thường ít tốn kém hơn so với việc thu hút khách hàng mới. Do đó, việc dự đoán và giảm thiểu tỷ lệ rời bỏ dịch vụ giúp công ty tiết kiệm chi phí marketing và quảng cáo.

Với những lý do trên, đề tài "Ứng dụng mô hình Super Learning để dự đoán xu hướng khách hàng rời bỏ dịch vụ bảo hiểm" là một đề tài có ý nghĩa thực tiễn cao, không chỉ giúp các công ty bảo hiểm nâng cao hiệu quả kinh doanh mà còn đóng góp vào sự phát triển của lĩnh vực bảo hiểm nói chung.

**1.1.2 Mục đích đề tài.**

Đề tài "Ứng dụng mô hình Super Learning để dự đoán xu hướng khách hàng rời bỏ dịch vụ bảo hiểm" nhằm đạt được các mục tiêu sau:

1. **Nghiên cứu và ứng dụng mô hình Super Learning**:

* Tìm hiểu về phương pháp Super Learning, các thành phần cấu thành và cách thức hoạt động.
* Xây dựng mô hình Super Learning phù hợp với dữ liệu khách hàng của công ty bảo hiểm.

1. **Dự đoán xu hướng khách hàng rời bỏ dịch vụ**:

* Sử dụng mô hình Super Learning để phân tích và dự đoán khả năng khách hàng rời bỏ dịch vụ bảo hiểm.
* Đánh giá độ chính xác và hiệu quả của mô hình trong việc dự đoán.

1. **Xác định các yếu tố ảnh hưởng đến quyết định rời bỏ dịch vụ của khách hàng**:

* Phân tích dữ liệu để xác định các yếu tố quan trọng như nhân khẩu học, thói quen sử dụng dịch vụ, phản hồi khách hàng, v.v.
* Xác định mối quan hệ giữa các yếu tố này và quyết định rời bỏ dịch vụ của khách hàng.

1. **Đưa ra các biện pháp giữ chân khách hàng**:

* Dựa trên kết quả dự đoán và phân tích, đề xuất các biện pháp nhằm giảm tỷ lệ rời bỏ dịch vụ.
* Xây dựng các chiến lược chăm sóc và tương tác với khách hàng tiềm năng có nguy cơ rời bỏ dịch vụ cao.

1. **Cải thiện chất lượng dịch vụ bảo hiểm**:

* Sử dụng thông tin từ mô hình dự đoán để nâng cao chất lượng dịch vụ và trải nghiệm khách hàng.
* Đưa ra các khuyến nghị về cải tiến sản phẩm, quy trình dịch vụ và hỗ trợ khách hàng.

1. **Nâng cao năng lực cạnh tranh của công ty bảo hiểm**:

* Tối ưu hóa chiến lược kinh doanh dựa trên các dự đoán và phân tích từ mô hình Super Learning.
* Giảm thiểu tổn thất tài chính do khách hàng rời bỏ dịch vụ và tăng cường lòng trung thành của khách hàng hiện tại.

Thông qua việc thực hiện đề tài này, công ty bảo hiểm có thể không chỉ dự đoán chính xác hơn về hành vi khách hàng mà còn triển khai các biện pháp hiệu quả để giữ chân khách hàng, cải thiện chất lượng dịch vụ và nâng cao năng lực cạnh tranh trên thị trường.

**1.1.3 Phướng pháp tiến hành.**

Bước 1: Tìm hiểu mô hình Ensemble Learning.

1. **Tìm hiểu lý thuyết và các kĩ thuật của Ensemble Learning**

* **Khái niệm Ensemble Learning**: Sử dụng nhiều thuật toán học tập để cải thiện hiệu suất dự đoán.
* **Lợi ích của Ensemble Learning**: Tăng độ chính xác, giảm overfitting, và nâng cao khả năng tổng quát hóa.
* **Nguyên lý hoạt động**: Kết hợp nhiều mô hình con để tạo ra một mô hình mạnh mẽ hơn.

1. **Tìm hiểu Basic Ensemble Techniques**

* **Max Voting**: Mô hình nào dự đoán nhiều nhất sẽ là kết quả cuối cùng.
* **Averaging**: Lấy trung bình dự đoán của các mô hình con.
* **Weighted Averaging**: Lấy trung bình dự đoán của các mô hình con, nhưng mỗi mô hình có trọng số khác nhau

1. **Tìm hiểu Advanced Ensemble Techniques**

* **Stacking (Stacked Generalization)**: Kết hợp dự đoán từ nhiều mô hình con thông qua một mô hình siêu (meta-model) để tạo ra dự đoán cuối cùng.
* **Bagging (Bootstrap Aggregating)**: Tạo nhiều mô hình con từ các tập con khác nhau của tập dữ liệu huấn luyện và kết hợp kết quả.
* **Blending**: Một biến thể của stacking, trong đó dữ liệu được chia thành hai tập, một để huấn luyện mô hình con và một để huấn luyện mô hình siêu.
* **Boosting**: Tăng cường trọng số cho các mẫu dữ liệu khó dự đoán và giảm trọng số cho các mẫu dễ dự đoán.

1. **Tìm hiểu cách Stacking được sử dụng trong Super Learning để kết hợp nhiều mô hình con**

* **Nguyên lý hoạt động của Stacking trong Super Learning**: Sử dụng nhiều mô hình con để tạo ra các dự đoán trung gian, sau đó kết hợp các dự đoán này bằng một mô hình siêu.
* **Ưu điểm của Stacking trong Super Learning**: Tận dụng sức mạnh của từng mô hình con, cải thiện độ chính xác tổng thể.

Bước 2: Tìm hiểu bài toán dự báo khách hàng rời bỏ dịch vụ bảo hiểm.

1. **Xác định vấn đề và mục tiêu**

* **Vấn đề chính**: Dự báo xu hướng khách hàng rời bỏ dịch vụ bảo hiểm.
* **Mục tiêu nghiên cứu**: Xây dựng mô hình dự báo chính xác xu hướng này để giúp công ty bảo hiểm giữ chân khách hàng.

1. **Nghiên cứu các yếu tố ảnh hưởng**

* **Phân tích các yếu tố ảnh hưởng**: Lịch sử sử dụng dịch vụ, thông tin nhân khẩu học, hành vi khách hàng, và các yếu tố kinh tế - xã hội khác.

1. **Xem xét các phương pháp dự báo hiện có**

* **Nghiên cứu các phương pháp dự báo hiện có**: Sử dụng các mô hình dự đoán như Logistic Regression, Decision Trees, và các phương pháp Ensemble khác.
* **So sánh với Super Learning**: Đánh giá lợi thế của Super Learning trong việc cải thiện độ chính xác dự đoán.

Bước 3: Tìm hiểu tập dữ liệu khách hàng sử dụng dịch vụ bảo hiểm.

1. **Thu thập dữ liệu**

* **Nguồn dữ liệu**: Dữ liệu từ các công ty bảo hiểm, dữ liệu công khai, và dữ liệu từ các cuộc khảo sát.
* **Loại dữ liệu**: Thông tin cá nhân, lịch sử giao dịch, thông tin hợp đồng bảo hiểm, và các thông tin liên quan khác.

1. **Xử lý dữ liệu**

* **Làm sạch dữ liệu**: Xử lý các giá trị thiếu, loại bỏ các giá trị bất thường, và chuẩn hóa dữ liệu.
* **Biến đổi dữ liệu**: Tạo các đặc trưng mới từ dữ liệu hiện có, chuyển đổi các biến số phân loại thành các biến số số học.

1. **Phân tích dữ liệu**

* **Phân tích mô tả**: Tóm tắt và trực quan hóa dữ liệu để hiểu rõ hơn về các đặc trưng và mối quan hệ giữa các biến số.
* **Phân tích thống kê**: Sử dụng các phương pháp thống kê để tìm hiểu các yếu tố có ảnh hưởng lớn đến việc rời bỏ dịch vụ.

Bước 4: Xây dựng mô hình phân loại khách hàng sử dụng dịch vụ bảo hiểm dựa trên mô hình Super Learning.

1. **Chuẩn bị dữ liệu cho mô hình**

* Tiền xử lý dữ liệu.
* **Tạo các đặc trưng (features)**: Chọn và tạo các đặc trưng phù hợp cho mô hình.
* **Chia dữ liệu**: Chia dữ liệu thành tập huấn luyện và tập kiểm tra.

1. **Huấn luyện các mô hình con**

* **Lựa chọn mô hình con**: Chọn các mô hình con phù hợp như Logistic Regression, Decision Trees, Random Forest, và Gradient Boosting, SVM.

1. **Huấn luyện mô hình siêu (meta-model)**

* **Tạo dự đoán trung gian**: Sử dụng các mô hình con để tạo ra các dự đoán trung gian trên tập dữ liệu huấn luyện.
* **Huấn luyện mô hình siêu**: Sử dụng các dự đoán trung gian làm đầu vào để huấn luyện mô hình siêu trên tập dữ liệu huấn luyện.

1. **Đánh giá hiệu suất mô hình**

* **Chỉ số đánh giá**: Sử dụng các chỉ số như độ chính xác, độ nhạy, độ đặc hiệu, và AUC-ROC để đánh giá hiệu suất của mô hình.

Bước 5: Xây dựng và triển khai ứng dụng dự báo khách hàng rời bỏ dịch vụ bảo hiểm.

1. **Tạo môi trường triển khai.**

* **Thiết lập môi trường**: Thiết lập môi trường tính toán và công cụ cần thiết để triển khai mô hình, như Python, Jupyter Notebook, và các thư viện học máy.
* **Chuẩn bị dữ liệu**: Chuẩn bị và cập nhật dữ liệu cần thiết cho ứng dụng.

1. **Xây dựng ứng dụng dự báo.**

* **Phát triển ứng dụng**: Phát triển ứng dụng hoặc hệ thống dự báo sử dụng mô hình Super Learning.
* **Tích hợp mô hình**: Tích hợp mô hình Super Learning vào hệ thống và thiết lập quy trình cập nhật dữ liệu và mô hình.

**CHƯƠNG 2: CƠ SỞ LÝ THUYẾT**

**2.1 Tổng quan về Ensemble Learning và Super Learning**

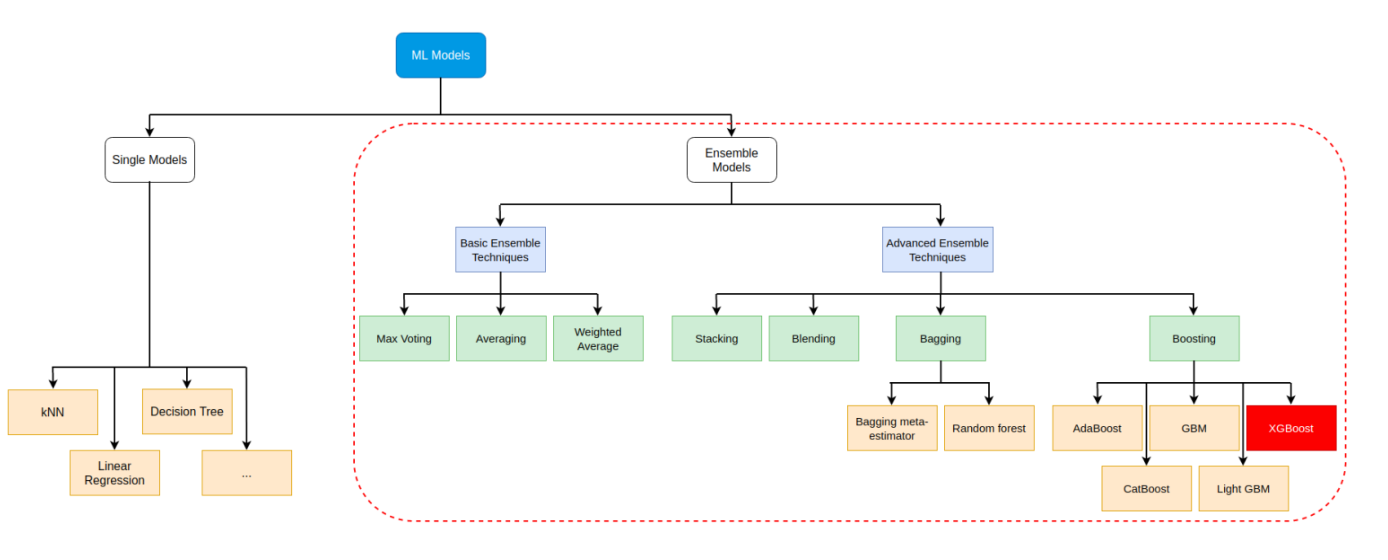
**2.1.1 Giới thiệu về Ensemble Learning.**

1. Khái niệm: là một kỹ thuật học máy kết hợp nhiều mô hình học máy khác nhau để tạo ra một mô hình dự đoán tốt hơn. Mỗi mô hình học tập từ một tập dữ liệu con hoặc sử dụng một thuật toán khác nhau, sau đó kết hợp dự đoán của các mô hình để đưa ra kết quả cuối cùng.
2. Lợi ích của Ensemble Learning:

* **Tăng độ chính xác**: Bằng cách kết hợp nhiều mô hình, Ensemble Learning có thể cải thiện độ chính xác của dự đoán.
* Giảm overfitting: Ensemble Learning giúp giảm thiểu hiện tượng overfitting bằng cách kết hợp các mô hình với nhau, làm giảm độ lệch và phương sai.
* Nâng cao khả năng tổng quát hóa: Các mô hình Ensemble thường tổng quát hóa tốt hơn trên dữ liệu chưa thấy.

- Có một số cách để cải thiện tính tổng quát hóa, nhưng có hai chính sách sau đây:

* Cải thiện hiệu suất của một máy học (model).
* Chuẩn bị nhiều máy học (model) và tổng hợp kết quả dự đoán của chúng để nâng cao tính tổng quát hóa.



*Hình 2.1.1 Sơ đồ tổng quan Ensemble Learning.*

**2.1.2 **Basic Ensemble Techniques.****

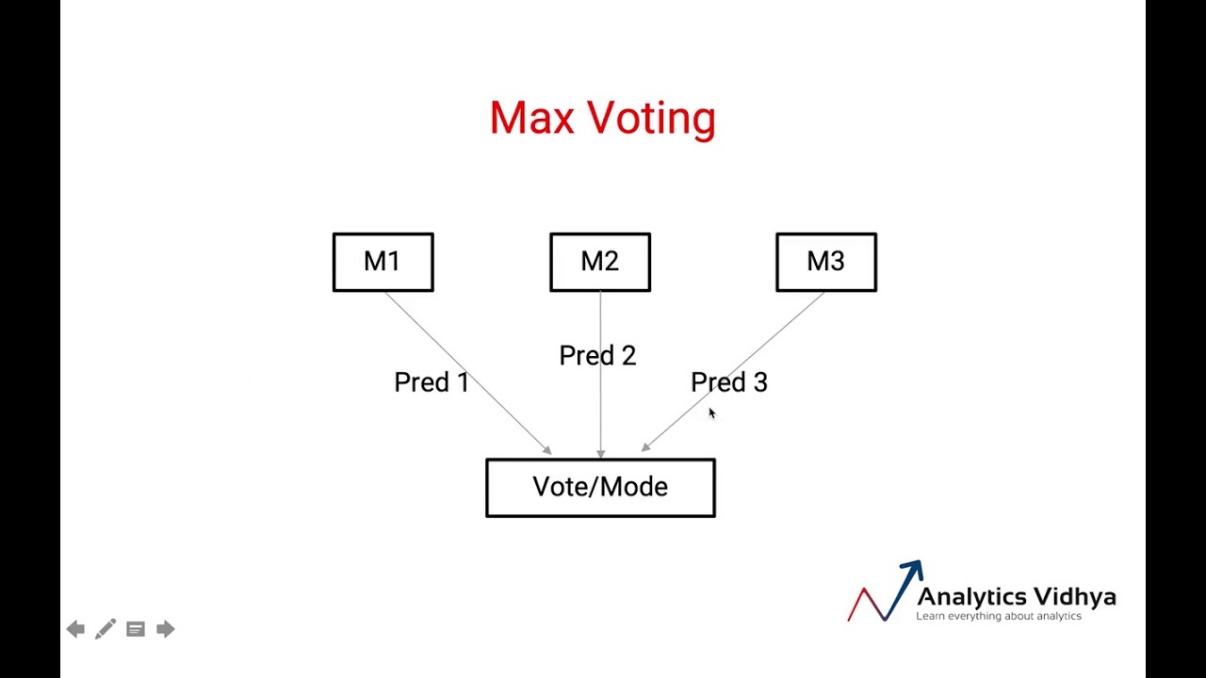
Ở mức độ cơ bản, có 3 kỹ thuật là:

* Max Voting
* Averaging
* Weighted Averaging

Mặc dù đơn giản nhưng những kỹ thuật này lại tỏ ra hiệu quả trong một số trường hợp nhất định. Hãy cùng tìm hiểu kỹ hơn về chúng.

1. Max Voting (bỏ phiêú đa số).

**Max Voting** là một kỹ thuật Ensemble Learning đơn giản nhưng hiệu quả, sử dụng phương pháp bỏ phiếu đa số để kết hợp dự đoán của các mô hình học máy khác nhau. Trong kỹ thuật này, mỗi mô hình học máy đưa ra dự đoán cho một mẫu dữ liệu, sau đó dự đoán được chọn nhiều nhất sẽ được coi là dự đoán cuối cùng cho mẫu dữ liệu đó.



*Hình ảnh 2.1.2 Sơ đồ* ***Max Voting.***

**Lý thuyết:**

Giả sử ta có N mô hình học máy và mỗi mô hình đưa ra dự đoán cho một mẫu dữ liệu x. Dự đoán của mô hình i được ký hiệu là yi​(x). Kỹ thuật Max Voting sẽ chọn dự đoán cuối cùng cho mẫu dữ liệu x là giá trị y có số lượng mô hình dự đoán cao nhất, được tính toán bằng công thức:

y(x)=argy∈Ymax​i=1∑N​I(yi​(x)=y)

Trong đó:

* Y là tập hợp các giá trị dự đoán có thể có.
* I(x) là hàm chỉ báo, trả về 1 nếu x đúng và 0 nếu x sai.

Lấy ví dụ, đợt vừa rồi, công ty của bạn tổ chức khám sức khỏe cho nhân viên tại bệnh viện X. Sau khi khám xong, phòng tổ chức nhân sự (TCNS) lấy ý kiến mọi người về chất lượng khám bệnh để xem năm sau có tiếp tục khám ở bênh viên X đó nữa không. Bảng dưới là ý kiến của 5 người được chọn ngẫu nhiên trong số toàn bộ nhân viên.

| **Người 1** | **Người 2** | **Người 3** | **Người 4** | **Người 5** |
| --- | --- | --- | --- | --- |
| Có | Không | Không | Có | Có |

Có 3 ý kiến muốn tiêp tục khám ở bệnh viện X vào năm sau, và 2 ý kiến muốn đổi bênh viện khác. Căn cứ theo max voting thì phòng TCNS sẽ tiếp tục chọn bệnh viên X là nơi khám bệnh cho nhân viên cho năm tiếp theo.

**Ưu điểm:**

* **Đơn giản:** Kỹ thuật Max Voting rất dễ hiểu và dễ triển khai.
* **Hiệu quả:** Kỹ thuật Max Voting có thể hiệu quả trong một số trường hợp, đặc biệt là khi các mô hình học máy có độ chính xác tương đương nhau.

**Nhược điểm:**

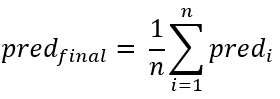
* **Không cân nhắc độ tin cậy của mô hình:** Kỹ thuật Max Voting không cân nhắc độ tin cậy của các mô hình học máy, tất cả các mô hình được coi là có độ tin cậy bằng nhau.
* **Có thể bị ảnh hưởng bởi các mô hình sai lệch:** Kỹ thuật Max Voting có thể bị ảnh hưởng bởi các mô hình sai lệch, dẫn đến kết quả dự đoán không chính xác.

#### Averaging (Trung bình cộng).

**Averaging** là một kỹ thuật Ensemble Learning khác, đơn giản nhưng hiệu quả, sử dụng phép tính trung bình cộng để kết hợp dự đoán của các mô hình học máy khác nhau. Trong kỹ thuật này, dự đoán của tất cả các mô hình học máy cho một mẫu dữ liệu được tính trung bình, và kết quả trung bình được coi là dự đoán cuối cùng.

**Lý thuyết:**

Giả sử ta có N mô hình học máy và mỗi mô hình đưa ra dự đoán cho một mẫu dữ liệu x. Dự đoán của mô hình i được ký hiệu là yi​(x). Kỹ thuật Averaging sẽ chọn dự đoán cuối cùng cho mẫu dữ liệu x là giá trị trung bình của các dự đoán của các mô hình, được tính toán bằng công thức:



**Lưu ý:**

* Kỹ thuật Averaging thường được sử dụng cho các bài toán hồi quy (regression), nơi dự đoán là một giá trị liên tục.
* Trong một số trường hợp, Averaging cũng có thể được sử dụng cho các bài toán phân loại (classification) bằng cách làm tròn giá trị trung bình về giá trị gần nhất.

**Ví dụ:**

Tiếp tục với ví dụ ở trên, một đề nghị khác của phòng TCNS là yêu cầu nhân viên chấm điểm chất lượng khám bệnh của bênh viện X, theo thang điểm từ 1 đến 5.

Bảng kết quả trả lời của 5 người ngẫu nhiên:

| **Người 1** | **Người 2** | **Người 3** | **Người 4** | **Người 5** |
| --- | --- | --- | --- | --- |
| 2 | 4 | 3 | 5 | 4 |

Điểm đánh giá cuối cùng sẽ là: **(2+4+3+5+4)/5 = 3.6**

**Ưu điểm:**

* **Đơn giản:** Kỹ thuật Averaging rất dễ hiểu và dễ dàng triển khai.
* **Hiệu quả:** Kỹ thuật Averaging có thể hiệu quả trong một số trường hợp, đặc biệt là khi các mô hình học máy có độ chính xác tương đương nhau và sai số của các mô hình có phân bố đối xứng.

**Nhược điểm:**

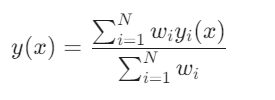
* **Không cân nhắc độ tin cậy của mô hình:** Kỹ thuật Averaging không cân nhắc độ tin cậy của các mô hình học máy, tất cả các mô hình được coi là có độ tin cậy bằng nhau.
* **Có thể bị ảnh hưởng bởi các mô hình ngoại lai:** Kỹ thuật Averaging có thể bị ảnh hưởng bởi các mô hình ngoại lai (outlier), do các mô hình này có thể làm sai lệch giá trị trung bình.

#### c ) Weighted Averaging (Trung bình cộng có trọng số)

**Weighted Averaging** là một kỹ thuật Ensemble Learning cải tiến so với Averaging, sử dụng phép tính trung bình cộng có trọng số để kết hợp dự đoán của các mô hình học máy khác nhau. Trong kỹ thuật này, mỗi mô hình học máy được gán một trọng số, phản ánh mức độ tin cậy của mô hình đó. Dự đoán của các mô hình sau đó được tính trung bình cộng với trọng số tương ứng, và kết quả trung bình có trọng số được coi là dự đoán cuối cùng.

**Lý thuyết:**

Giả sử ta có N mô hình học máy và mỗi mô hình đưa ra dự đoán cho một mẫu dữ liệu x. Dự đoán của mô hình i được ký hiệu là yi(x). Trọng số của mô hình i được ký hiệu là wi. Kỹ thuật Weighted Averaging sẽ chọn dự đoán cuối cùng cho mẫu dữ liệu x là giá trị trung bình có trọng số của các dự đoán của các mô hình, được tính toán bằng công thức:



**Trong đó:**

* wi​ là trọng số của mô hình i, với 0≤wi≤1 và 

Vẫn với ví dụ ở mục b) , nhưng trong số 5 người được hỏi thì người thứ nhất có vợ là bác sĩ, người thứ 2 có mẹ là y tá, người thứ 3 có người yêu là sinh viên trường y. Vì vậy, ý kiến của 3 người này rõ ràng có giá trị hơn so với 2 người còn lại.  
Ta đánh trọng số cho mỗi người như bảng dưới (*hàng thứ 2 là trọng số, hàng thứ 3 là điểm đánh giá*):

| **Người 1** | **Người 2** | **Người 3** | **Người 4** | **Người 5** |
| --- | --- | --- | --- | --- |
| 1 | 0.8 | 0.5 | 0.3 | 0.3 |
| 2 | 4 | 3 | 5 | 4 |

Điểm đánh giá cuối cùng sẽ là: \**(2*1 + 4*0.8 + 3*0.5 + 5*0.3 + 4*0.3)/5 = 1.88\*\*

**Ưu điểm:**

* **Cân nhắc độ tin cậy của mô hình:** Kỹ thuật Weighted Averaging cho phép cân nhắc độ tin cậy của các mô hình học máy bằng cách sử dụng trọng số.
* **Có thể hiệu quả hơn Averaging:** Khi sử dụng trọng số phù hợp, Weighted Averaging có thể dẫn đến kết quả dự đoán chính xác hơn so với Averaging.

**Nhược điểm:**

* **Phụ thuộc vào việc xác định trọng số:** Hiệu quả của Weighted Averaging phụ thuộc nhiều vào việc xác định chính xác trọng số của các mô hình.
* **Có thể phức tạp hơn Averaging:** Xác định trọng số của các mô hình có thể phức tạp hơn so với Averaging.

**2.1.3 **Advanced Ensemble techniques.****

Có 4 kỹ thuật của Ensemble Learning được xếp vào nhóm advanced:

* Bagging
* Boosting
* Blending
* Stacking

Chúng ta tiếp tục đi qua lần lượt từng kỹ thuật này:

#### a ) Bagging (Bootstrap Aggregating).

Bagging (Bootstrap Aggregating) là một kỹ thuật học tập tập thể (Ensemble Learning) sử dụng nhiều mô hình học máy được huấn luyện trên các tập dữ liệu con lấy mẫu ngẫu nhiên từ tập dữ liệu gốc với thay thế. Sau đó, dự đoán cuối cùng cho một điểm dữ liệu mới được đưa ra bằng cách kết hợp dự đoán từ các mô hình con (thường bằng cách trung bình cộng).

Lý thuyết:

* **Lấy mẫu ngẫu nhiên với thay thế:** Từ tập dữ liệu gốc, ta tạo ra nhiều tập dữ liệu con (gọi là "túi" - "bag") bằng cách lấy mẫu ngẫu nhiên với thay thế. Mỗi túi có kích thước bằng kích thước tập dữ liệu gốc, nhưng có thể chứa một số điểm dữ liệu trùng lặp.
* **Huấn luyện mô hình con:** Trên mỗi tập dữ liệu con, ta huấn luyện một mô hình học máy riêng biệt (gọi là "mô hình con"). Các mô hình con thường sử dụng cùng một thuật toán học máy, nhưng có thể có các tham số khác nhau.
* **Kết hợp dự đoán:** Khi có một điểm dữ liệu mới cần dự đoán, ta cho mỗi mô hình con dự đoán điểm dữ liệu đó. Dự đoán cuối cùng được đưa ra bằng cách kết hợp dự đoán từ các mô hình con (thường bằng cách trung bình cộng).

**Ưu điểm:**

* **Giảm phương sai:** Bagging giúp giảm phương sai của mô hình học máy, do đó làm cho dự đoán ổn định hơn và ít bị ảnh hưởng bởi nhiễu trong dữ liệu.
* **Chống overfitting:** Do các mô hình con được huấn luyện trên các tập dữ liệu con khác nhau, nên chúng ít có khả năng bị overfitting hơn so với mô hình được huấn luyện trên toàn bộ tập dữ liệu gốc.
* **Có thể áp dụng cho nhiều thuật toán học máy:** Bagging có thể được áp dụng cho nhiều thuật toán học máy khác nhau, bao gồm cây quyết định, mạng nơ-ron nhân tạo, v.v.

**Nhược điểm:**

* **Tăng chi phí tính toán:** Do cần huấn luyện nhiều mô hình con, bagging có thể tốn nhiều thời gian và tài nguyên tính toán hơn so với các phương pháp học máy khác.
* **Có thể giảm hiệu suất nếu độ tương quan giữa các mô hình con cao:** Nếu các mô hình con có dự đoán rất giống nhau, thì việc kết hợp dự đoán của chúng có thể không mang lại nhiều lợi ích.

b ) Boosting.

Boosting là một kỹ thuật học tập tập thể (Ensemble Learning) khác thường được so sánh với Bagging. Giống như Bagging, Boosting cũng kết hợp nhiều mô hình học máy để tạo ra một mô hình mạnh hơn. Tuy nhiên, cách thức hoạt động của Boosting lại khác biệt.

Lý thuyết:

* **Khởi tạo mô hình yếu:** Boosting bắt đầu bằng việc huấn luyện một mô hình yếu (thường là một mô hình học máy đơn giản) trên toàn bộ tập dữ liệu gốc.
* **Phân tích lỗi:** Dự đoán của mô hình yếu được so sánh với nhãn thực tế của dữ liệu. Dữ liệu được phân thành hai nhóm: nhóm các điểm dữ liệu được mô hình dự đoán chính xác và nhóm các điểm dữ liệu được mô hình dự đoán sai.
* **Tạo trọng số:** Sau đó, Boosting gán trọng số cho mỗi điểm dữ liệu. Các điểm dữ liệu được mô hình dự đoán sai sẽ được gán trọng số cao hơn, nghĩa là chúng sẽ có ảnh hưởng lớn hơn đến việc huấn luyện mô hình tiếp theo.
* **Huấn luyện mô hình tiếp theo:** Mô hình tiếp theo được huấn luyện trên toàn bộ tập dữ liệu gốc, nhưng với trọng số được gán cho từng điểm dữ liệu. Mục tiêu của mô hình này là cải thiện hiệu suất trên các điểm dữ liệu mà mô hình trước đó dự đoán sai.
* **Lặp lại các bước:** Bước 2-4 được lặp lại nhiều lần. Tại mỗi lần lặp, mô hình mới được huấn luyện dựa trên trọng số được cập nhật, giúp cải thiện dần dần hiệu suất tổng thể của mô hình ensemble.
* **Kết hợp dự đoán:** Cuối cùng, dự đoán cuối cùng cho một điểm dữ liệu mới được đưa ra bằng cách kết hợp dự đoán từ các mô hình con theo một thuật toán nhất định (ví dụ như weighted voting).

**Ưu điểm:**

* **Giảm cả phương sai và bias:** Boosting có thể giúp giảm cả phương sai và bias của mô hình học máy, do đó thường dẫn đến hiệu suất tốt hơn so với Bagging.
* **Có thể xử lý các bài toán phân loại khó:** Boosting hoạt động hiệu quả với các bài toán phân loại khó, nơi mà ranh giới giữa các lớp rất phức tạp.

**Nhược điểm:**

* **Dễ bị overfitting:** Boosting có thể dễ bị overfitting hơn so với Bagging, đặc biệt nếu sử dụng quá nhiều mô hình con.
* **Có thể nhạy cảm với nhiễu trong dữ liệu:** Nếu dữ liệu bị nhiễu, Boosting có thể học các mô hình yếu tập trung vào nhiễu thay vì các mẫu hình thực tế trong dữ liệu.
* **Yêu cầu nhiều thời gian tính toán:** Cũng giống như Bagging, Boosting cần huấn luyện nhiều mô hình con, nên có thể tốn nhiều thời gian và tài nguyên tính toán hơn so với các phương pháp học máy khác.

c ) Blending.

Blending, hay còn gọi là Stacked Generalization, là một kỹ thuật học máy tập thể (Ensemble Learning) cùng với Bagging và Boosting. Kỹ thuật này tập trung vào việc kết hợp dự đoán từ các mô hình cơ sở (base model) khác nhau để tạo ra dự đoán cuối cùng, tiềm năng mang lại độ chính xác cao hơn.

Lý thuyết:

* **Chia tách dữ liệu:** Tập dữ liệu gốc được chia thành hai phần: tập huấn luyện và tập xác nhận.
* **Huấn luyện các mô hình cơ sở:** Nhiều mô hình cơ sở (các thuật toán học máy khác nhau) được huấn luyện trên tập huấn luyện.
* **Tạo dự đoán:** Cả tập huấn luyện và tập xác nhận được sử dụng để lấy dự đoán từ mỗi mô hình cơ sở.
* **Chuẩn bị các siêu đặc trưng:** Dự đoán từ tập xác nhận cho mỗi mô hình cơ sở trở thành các đặc trưng mới, tạo thành một "siêu tập dữ liệu."
* **Huấn luyện mô hình meta:** Một mô hình mới, gọi là mô hình meta, được huấn luyện trên siêu tập dữ liệu này. Mô hình này học cách kết hợp tốt nhất các dự đoán từ các mô hình cơ sở.
* **Dự đoán cuối cùng:** Khi có dữ liệu mới, các mô hình cơ sở sẽ thực hiện dự đoán trước. Các dự đoán này sau đó được đưa vào mô hình meta, tạo ra dự đoán cuối cùng.

**Ưu điểm:**

* **Hiệu suất được cải thiện:** Bằng cách kết hợp sức mạnh của nhiều mô hình, Blending có thể tiềm năng dẫn đến dự đoán chính xác hơn so với các mô hình cơ sở riêng lẻ.
* **Giảm thiểu hiện tượng quá khớp (Overfitting):** Mô hình meta học cách tận dụng thông tin bổ sung từ các mô hình cơ sở đa dạng, tiềm năng giảm thiểu hiện tượng quá khớp trên dữ liệu huấn luyện.

**Nhược điểm:**

* **Độ phức tạp cao hơn:** Blending liên quan đến việc huấn luyện nhiều mô hình và đòi hỏi phải lựa chọn cẩn thận cả mô hình cơ sở và mô hình meta. Điều này có thể phức tạp hơn so với việc sử dụng một mô hình duy nhất.
* **Chi phí tính toán cao:** Việc huấn luyện nhiều mô hình có thể tốn kém về mặt tính toán, đặc biệt khi xử lý tập dữ liệu lớn.

d ) Stacking.

Stacking là một kỹ thuật học máy tập thể (Ensemble Learning) sử dụng nhiều mô hình học máy được huấn luyện ở các cấp độ khác nhau để tạo ra dự đoán cuối cùng. Khác với Bagging và Boosting, Stacking kết hợp các mô hình ở các cấp độ khác nhau thay vì chỉ kết hợp các mô hình cùng cấp.

Lý thuyết:

* **Cấp độ 1:** Huấn luyện nhiều mô hình học máy cơ sở (base model) trên tập dữ liệu gốc.
* **Dự đoán cấp độ 1:** Mỗi mô hình cơ sở tạo ra dự đoán cho các điểm dữ liệu trong tập dữ liệu gốc.
* **Tạo tập dữ liệu meta:** Kết hợp các dự đoán từ các mô hình cơ sở thành các đặc trưng mới cho mỗi điểm dữ liệu. Các đặc trưng này cùng với nhãn thực tế của dữ liệu tạo thành tập dữ liệu meta.
* **Huấn luyện mô hình meta:** Huấn luyện một mô hình học máy mới (mô hình meta) trên tập dữ liệu meta. Mô hình meta học cách kết hợp các dự đoán từ các mô hình cơ sở để tạo ra dự đoán cuối cùng chính xác hơn.
* **Dự đoán cuối cùng:** Khi có dữ liệu mới, các mô hình cơ sở tạo ra dự đoán cho điểm dữ liệu mới. Các dự đoán này được đưa vào mô hình meta để tạo ra dự đoán cuối cùng.

**Ưu điểm:**

* **Cải thiện hiệu suất dự đoán:** Stacking có thể tiềm năng đạt được độ chính xác cao hơn so với các mô hình học máy riêng lẻ.
* **Giảm thiểu lỗi:** Mô hình meta học cách kết hợp các dự đoán từ các mô hình cơ sở, giúp giảm thiểu lỗi và cải thiện độ chính xác.
* **Có thể áp dụng cho nhiều nhiệm vụ:** Stacking có thể được áp dụng cho nhiều loại nhiệm vụ học máy, bao gồm phân loại, hồi quy và xử lý ngôn ngữ tự nhiên.

**Nhược điểm:**

* **Phức tạp hơn:** Stacking có thể phức tạp hơn so với các kỹ thuật học máy tập thể khác như Bagging và Boosting.
* **Yêu cầu nhiều dữ liệu:** Stacking cần nhiều dữ liệu để huấn luyện cả mô hình cơ sở và mô hình meta.
* **Có thể tốn thời gian:** Huấn luyện nhiều mô hình học máy có thể tốn thời gian.

**2.1.4 Stacking được sử dụng trong Super Learning để kết hợp nhiều mô hình con.**

Nguyên lý hoạt động của Stacking trong Super Learning

**Bước 1: Huấn luyện các mô hình con:**

* Nhiều mô hình con (thường là các thuật toán học máy khác nhau) được huấn luyện độc lập trên cùng một tập dữ liệu.
* Mỗi mô hình con tạo ra một tập dự đoán cho các điểm dữ liệu trong tập huấn luyện.

**Bước 2: Tạo tập dữ liệu meta:**

* Các dự đoán từ các mô hình con được kết hợp thành các đặc trưng mới cho mỗi điểm dữ liệu.
* Các đặc trưng này cùng với nhãn thực tế của dữ liệu tạo thành tập dữ liệu meta.

**Bước 3: Huấn luyện mô hình siêu:**

* Một mô hình học máy mới (mô hình siêu) được huấn luyện trên tập dữ liệu meta.
* Mô hình siêu học cách kết hợp các dự đoán từ các mô hình con một cách hiệu quả để tạo ra dự đoán cuối cùng chính xác hơn.

**Bước 4: Dự đoán cho dữ liệu mới:**

* Khi có dữ liệu mới, các mô hình con tạo ra dự đoán cho dữ liệu đó.
* Các dự đoán này được đưa vào mô hình siêu để tạo ra dự đoán cuối cùng.

**Ưu điểm:**

* **Tận dụng sức mạnh của từng mô hình con:** Stacking cho phép mỗi mô hình con phát huy thế mạnh riêng, góp phần cải thiện hiệu suất tổng thể.
* **Cải thiện độ chính xác:** Bằng cách kết hợp các dự đoán từ nhiều mô hình con, Stacking có thể giảm thiểu lỗi và cải thiện độ chính xác dự đoán, đặc biệt khi các mô hình con có những điểm mạnh và điểm yếu khác nhau.
* **Giảm thiểu Overfitting:** Mô hình siêu học cách kết hợp các dự đoán từ các mô hình con, giúp giảm thiểu ảnh hưởng của Overfitting trên từng mô hình con riêng lẻ.
* **Linh hoạt:** Stacking có thể được áp dụng cho nhiều loại nhiệm vụ học máy, bao gồm phân loại, hồi quy và xử lý ngôn ngữ tự nhiên.

**2.2 Tìm hiểu bài toán dự báo khách hàng rời bỏ dịch vụ bảo hiểm.**

****2.2.1 Xác định vấn đề và mục tiêu.****

a ) Vấn đề:

Dự báo xu hướng khách hàng rời bỏ dịch vụ bảo hiểm là một thách thức quan trọng đối với các công ty bảo hiểm. Khả năng dự đoán chính xác hành vi của khách hàng giúp doanh nghiệp chủ động triển khai các biện pháp giữ chân khách hàng tiềm năng, giảm thiểu tổn thất tài chính và nâng cao hiệu quả hoạt động.

**Hậu quả của việc không dự đoán chính xác:**

* **Mất khách hàng:** Khách hàng không hài lòng hoặc không nhận được giá trị từ dịch vụ có thể hủy hợp đồng, dẫn đến doanh thu giảm và lợi nhuận thấp.
* **Tăng chi phí tiếp thị:** Việc thu hút khách hàng mới tốn kém hơn so với việc giữ chân khách hàng hiện tại.
* **Tổn hại uy tín thương hiệu:** Khách hàng chia sẻ trải nghiệm tiêu cực có thể ảnh hưởng đến danh tiếng của công ty và thu hút ít khách hàng tiềm năng hơn.
* **Thiếu hiệu quả hoạt động:** Việc không dự đoán được tỷ lệ khách hàng rời bỏ dịch vụ có thể dẫn đến việc quản lý tài chính và nguồn lực không hiệu quả.

b ) **Mục tiêu:**

Mục tiêu của bài toán dự báo khách hàng rời bỏ dịch vụ bảo hiểm là xây dựng mô hình dự đoán chính xác xu hướng này để giúp công ty bảo hiểm:

* **Xác định những khách hàng có nguy cơ cao rời bỏ dịch vụ:** Nhờ vậy, công ty có thể tập trung nguồn lực và thực hiện các biện pháp can thiệp kịp thời để giữ chân những khách hàng quan trọng.
* **Hiểu rõ hơn về các yếu tố ảnh hưởng đến hành vi khách hàng:** Việc phân tích các yếu tố này giúp công ty cải thiện chất lượng dịch vụ, sản phẩm và chiến lược marketing để đáp ứng tốt hơn nhu cầu khách hàng.
* **Tăng cường hiệu quả hoạt động:** Dự báo chính xác tỷ lệ khách hàng rời bỏ dịch vụ giúp công ty dự trù ngân sách, quản lý rủi ro và tối ưu hóa chiến lược kinh doanh.

****2.2.2** Nghiên cứu các yếu tố ảnh hưởng đến hành vi rời bỏ dịch vụ bảo hiểm.**

**Phân tích các yếu tố ảnh hưởng:**

Để xây dựng mô hình dự báo khách hàng rời bỏ dịch vụ bảo hiểm chính xác, cần phân tích kỹ lưỡng các yếu tố ảnh hưởng đến hành vi này. Các yếu tố này có thể được chia thành 4 nhóm chính:

1. **Lịch sử sử dụng dịch vụ:**

* **Thời gian sử dụng dịch vụ:** Khách hàng sử dụng dịch vụ càng lâu, khả năng trung thành càng cao và ít có nguy cơ rời bỏ dịch vụ hơn.
* **Tần suất sử dụng dịch vụ:** Khách hàng sử dụng dịch vụ thường xuyên cho thấy mức độ hài lòng cao và nhu cầu lớn, do đó ít có khả năng rời bỏ dịch vụ.
* **Số lượng hợp đồng:** Khách hàng có nhiều hợp đồng bảo hiểm với công ty có khả năng gắn kết cao hơn và ít có nguy cơ hủy hợp đồng.
* **Mức phí bảo hiểm:** Mức phí bảo hiểm cao có thể khiến khách hàng cân nhắc tìm kiếm các lựa chọn thay thế rẻ hơn, dẫn đến nguy cơ rời bỏ dịch vụ.
* **Số lần bồi thường:** Khách hàng có nhiều lần bồi thường có thể cho thấy mức độ rủi ro cao hơn, dẫn đến khả năng điều chỉnh phí bảo hiểm hoặc hủy hợp đồng.
* **Mức độ hài lòng với dịch vụ:** Khách hàng không hài lòng với dịch vụ, thủ tục bồi thường phức tạp, hoặc chất lượng chăm sóc khách hàng kém có thể dẫn đến việc hủy hợp đồng.

1. **Thông tin nhân khẩu học:**

* **Tuổi tác:** Khách hàng trẻ tuổi có xu hướng di chuyển nhiều hơn và thay đổi nhu cầu bảo hiểm, dẫn đến khả năng hủy hợp đồng cao hơn.
* **Giới tính:** Nam giới và nữ giới có thể có mức độ rủi ro và nhu cầu bảo hiểm khác nhau, ảnh hưởng đến hành vi rời bỏ dịch vụ.
* **Nghề nghiệp:** Nghề nghiệp có nguy cơ cao có thể dẫn đến nhu cầu bảo hiểm cao hơn hoặc thay đổi rủi ro, ảnh hưởng đến quyết định giữ chân dịch vụ.
* **Thu nhập:** Khách hàng có thu nhập cao có khả năng chi trả cho các dịch vụ bảo hiểm tốt hơn, nhưng cũng có thể có nhiều lựa chọn thay thế, dẫn đến khả năng rời bỏ dịch vụ cao hơn.
* **Tình trạng hôn nhân:** Khách hàng đã kết hôn hoặc có gia đình có thể có nhu cầu bảo hiểm cao hơn và ít có khả năng hủy hợp đồng hơn.
* **Trình độ học vấn:** Khách hàng có trình độ học vấn cao hơn có thể có nhận thức tốt hơn về tầm quan trọng của bảo hiểm và ít có khả năng hủy hợp đồng hơn.

1. **Hành vi khách hàng:**

* **Mức độ tương tác với công ty:** Khách hàng thường xuyên tương tác với công ty qua website, ứng dụng di động, hoặc gọi điện thoại có thể cho thấy mức độ gắn kết cao và ít có khả năng hủy hợp đồng.
* **Phản hồi trên mạng xã hội:** Khách hàng thường xuyên chia sẻ phản hồi tích cực về công ty trên mạng xã hội có thể cho thấy mức độ hài lòng cao và ít có khả năng hủy hợp đồng.
* **Kháng nghị:** Khách hàng có nhiều khiếu nại hoặc yêu cầu bồi thường có thể cho thấy mức độ không hài lòng với dịch vụ và có khả năng hủy hợp đồng cao hơn.

1. **Yếu tố kinh tế - xã hội:**

* **Tình hình kinh tế chung:** Tình hình kinh tế khó khăn có thể khiến khách hàng cắt giảm chi tiêu cho các dịch vụ không thiết yếu như bảo hiểm, dẫn đến khả năng hủy hợp đồng cao hơn.
* **Tỷ lệ thất nghiệp:** Tỷ lệ thất nghiệp cao có thể ảnh hưởng đến khả năng chi trả của khách hàng cho dịch vụ bảo hiểm, dẫn đến khả năng hủy hợp đồng cao hơn.
* **Lạm phát:** Lạm phát cao có thể khiến giá cả dịch vụ tăng lên, dẫn đến gánh nặng chi phí cho khách hàng và khả năng hủy hợp đồng cao hơn.
* **Cạnh tranh:** Mức độ cạnh tranh cao trong ngành bảo hiểm có thể khiến khách hàng dễ dàng chuyển sang các nhà cung cấp khác, dẫn đến khả năng hủy hợp đồng cao hơn.
* **Chính sách của chính phủ:** Các thay đổi về chính sách thuế, quy định bảo hiểm, hoặc các chương trình hỗ trợ tài chính có thể ảnh hưởng đến nhu cầu và hành vi của khách hàng, dẫn đến khả năng hủy hợp đồng.

**2.2.3 Xem xét các phương pháp dự báo hiện có.**

Có nhiều phương pháp dự báo khác nhau có thể được sử dụng để dự đoán xu hướng khách hàng rời bỏ dịch vụ bảo hiểm. Một số phương pháp phổ biến bao gồm:

* **Hồi quy logistic:** Đây là một mô hình thống kê được sử dụng để dự đoán khả năng xảy ra một sự kiện nhị phân, chẳng hạn như khách hàng có rời bỏ dịch vụ bảo hiểm hay không.
* **Cây quyết định:** Cây quyết định là một mô hình học máy sử dụng một loạt các quy tắc để phân loại dữ liệu. Mô hình này có thể được sử dụng để xác định các yếu tố chính ảnh hưởng đến việc khách hàng rời bỏ dịch vụ bảo hiểm.
* **Máy vectơ hỗ trợ (SVM):** SVM là một thuật toán học máy được sử dụng để phân loại dữ liệu và có thể được sử dụng để dự đoán khách hàng có khả năng rời bỏ dịch vụ bảo hiểm hay không.

Super Learning là một phương pháp học máy kết hợp nhiều mô hình dự đoán khác nhau để tạo ra một mô hình dự đoán chính xác hơn. Super Learning có thể được sử dụng để dự đoán xu hướng khách hàng rời bỏ dịch vụ bảo hiểm bằng cách kết hợp các mô hình dự đoán khác nhau như hồi quy logistic, cây quyết định và SVM.

Lợi thế chính của Super Learning là nó có thể cải thiện độ chính xác dự đoán bằng cách kết hợp các điểm mạnh của các mô hình dự đoán khác nhau. Super Learning cũng có thể giúp giảm thiểu nguy cơ quá khớp, xảy ra khi một mô hình dự đoán học quá tốt trên dữ liệu đào tạo nhưng không thể dự đoán chính xác trên dữ liệu mới.

**2.3 Tìm hiểu tập dữ liệu khách hàng sử dụng dịch vụ bảo hiểm.**

**2.3.1 Thu thập dữ liệu.**

**Nguồn dữ liệu:**

* **Dữ liệu từ các công ty bảo hiểm:** Bao gồm thông tin về khách hàng, lịch sử giao dịch, thông tin hợp đồng bảo hiểm và các thông tin liên quan khác. Có thể truy cập dữ liệu này thông qua các kênh như API, cơ sở dữ liệu khách hàng hoặc các hợp đồng chia sẻ dữ liệu.
* **Dữ liệu công khai:** Bao gồm thông tin về nhân khẩu học, kinh tế, xã hội và các yếu tố khác có thể ảnh hưởng đến việc sử dụng dịch vụ bảo hiểm. Dữ liệu này có thể thu thập từ các trang web của chính phủ, các tổ chức thống kê và các nguồn khác.
* **Dữ liệu từ các cuộc khảo sát:** Thu thập thông tin trực tiếp từ khách hàng về nhu cầu, mong muốn và trải nghiệm của họ với dịch vụ bảo hiểm. Các cuộc khảo sát có thể được thực hiện trực tuyến, qua điện thoại hoặc qua thư.
* Dữ liệu từ các trang web chuyên về dataset như kaggle, data.world, KDNuggets, Google Dataset Search. Ví dụ: <https://www.kaggle.com/datasets/mukulsingh/insurance-churn-prediction> (dataset mẫu về dự đoán tỉ lệ bảo hiểm) .

**Loại dữ liệu:**

* **Thông tin cá nhân:** Tên tuổi, địa chỉ, số điện thoại, ngày sinh, giới tính, v.v.
* **Lịch sử giao dịch:** Ngày mua bảo hiểm, loại bảo hiểm, mức phí bảo hiểm, số tiền bồi thường đã nhận, v.v.
* **Thông tin hợp đồng bảo hiểm:** Loại hợp đồng, thời hạn hiệu lực, phạm vi bảo hiểm, các điều khoản và điều kiện, v.v.
* **Các thông tin liên quan khác:** Nghề nghiệp, thu nhập, trình độ học vấn, tình trạng hôn nhân, sở thích, v.v.

**2.3.2 Xử lý dữ liệu.**

**Làm sạch dữ liệu: Xác định giá trị thiếu:** Kiểm tra xem có giá trị thiếu (null) trong bất kỳ cột nào của bảng dữ liệu không. Bạn có thể sử dụng phương thức isnull() trong thư viện Pandas để kiểm tra này.

**Xử lý giá trị thiếu:**

* 1. **Loại bỏ các hàng:** Nếu có rất ít giá trị thiếu và chúng phân bố ngẫu nhiên, bạn có thể loại bỏ các hàng chứa giá trị thiếu.
  2. **Điền giá trị trung bình:** Đối với các biến số liên tục (ví dụ: Claim Amount, BMI), có thể thay thế giá trị thiếu bằng giá trị trung bình của cột đó.
  3. **Điền giá trị dựa trên nhóm:** Đối với các biến phân loại (ví dụ: Claim Reason), có thể thay thế giá trị thiếu bằng giá trị phổ biến nhất trong nhóm (ví dụ: theo hạng mục phí bảo hiểm Category Premium).

**Loại bỏ giá trị bất thường:**

* 1. **Phương pháp IQR:** Xác định các ngoại lai (outlier) bằng cách tính IQR (Interquartile Range) cho các biến số liên tục. Loại bỏ các giá trị nằm ngoài khoảng IQR \* 1.5 dưới Q1 (giải tứ phân vị thứ nhất) và trên Q3 (giải tứ phân vị thứ ba).
  2. **Phương pháp z-score:** Chuẩn hóa các biến liên tục thành điểm chuẩn (z-score). Loại bỏ các giá trị có z-score lớn hơn hoặc nhỏ hơn một ngưỡng nhất định.

**Biến đổi dữ liệu:**

1. **Tạo các đặc trưng mới:**

* **Tuổi:** Có thể tính toán tuổi của khách hàng dựa trên ngày sinh (nếu có) trong biến Customer Name.
* **Số năm tham gia bảo hiểm:** Tính toán số năm khách hàng đã tham gia bảo hiểm dựa trên ngày mua bảo hiểm (có thể giả định từ biến Claim Reason).
* **Tổng số tiền yêu cầu bồi thường:** Tính tổng số tiền yêu cầu bồi thường đã được yêu cầu bởi mỗi khách hàng.

1. **Chuyển đổi biến số phân loại thành biến số số học:**

* **Mã hóa các biến phân loại:**
* Biến Category Premium: Gán số thứ tự cho mỗi hạng mục phí bảo hiểm (ví dụ: "Travel" = 1, "Medical" = 2, v.v.).
* Biến Data confidentiality: Gán số cho các mức độ bảo mật dữ liệu (ví dụ: "Low" = 0, "Medium" = 1, "High" = 2).
* Biến Claim Request output: Gán số cho kết quả yêu cầu bồi thường (ví dụ: "No" = 0, "Yes" = 1, "Pending" = 2).
* **Biến** Churn**:** Biến này đã được mã hóa sẵn (Yes = 1, No = 0).

**2.4 **Phân tích dữ liệu.****

**2.4.1 Phân tích mô tả:**

* Sử dụng hàm describe() của Pandas để tính toán các thống kê cơ bản cho các biến số số (numerical): trung bình, độ lệch chuẩn, giá trị tối thiểu, giá trị tối đa.
* Sử dụng phương thức value\_counts() để đếm số lần xuất hiện của mỗi giá trị phân loại (categorical).

|  |
| --- |
| data = pd.read\_csv('randomdata.csv')  num\_columns = data.shape[1]  print(f'Số cột: {num\_columns}')  print(data.describe())  # Số lần xuất hiện của các giá trị phân loại trong biến "Churn"  print(data["Churn"].value\_counts())  # Số lần xuất hiện của các giá trị phân loại trong biến "Claim Reason"  print(data["Claim Reason"].value\_counts()) |

*Hình 2.4.1 Code python thống kê cơ bản cho các biến số dữ liệu.*

Và kết quả :

|  |
| --- |
|  |

*Hình 2.4.2 Kết quả sử dụng hàm describe để thống kê.*

Giải thích:

* Tính toán thống kê cơ bản: Sử dụng hàm describe() của Pandas để tính toán các thông số cơ bản cho các biến số số, bao gồm : Giá trị trung bình (Mean), độ lệch chuẩn (Std), giá trị tối thiểu (Min), giá trị tối đa (Max), các phân vị (Quartiles).
* Đếm số lần xuất hiện của các giá trị phân loại: Sử dụng phương thức value\_counts() để đếm số lần mỗi giá trị phân loại xuất hiện trong dữ liệu.

**2.4.2 Phân tích theo nhóm:**

* Sử dụng phương thức groupby() để nhóm dữ liệu theo các biến phân loại (ví dụ: Category Premium, BMI, độ tuổi).
* Tính toán các thống kê mô tả (trung bình, tỷ lệ) cho biến Churn theo từng nhóm.

|  |
| --- |
| # Chuyển đổi cột Churn sang kiểu số (0 và 1)  data['Churn'] = data['Churn'].apply(lambda x: 1 if x == 'Yes' else 0)  # Tính toán độ tuổi từ ngày sinh (nếu có)  if 'Birthdate' in data.columns:      current\_date = datetime.now()      data['Birthdate'] = pd.to\_datetime(data['Birthdate'])      data['Age'] = data['Birthdate'].apply(lambda x: current\_date.year - x.year - ((current\_date.month, current\_date.day) < (x.month, x.day)))  else:      print("Không có cột 'Birthdate' trong dữ liệu.")  # Tỷ lệ hủy dịch vụ theo hạng mục phí bảo hiểm  print("Tỷ lệ hủy dịch vụ theo hạng mục phí bảo hiểm:")  print(data.groupby("Category Premium")["Churn"].mean())  # Tỷ lệ hủy dịch vụ theo BMI (cân nặng)  print("Tỷ lệ hủy dịch vụ theo BMI:")  print(data.groupby("BMI")["Churn"].mean())  # Tỷ lệ hủy dịch vụ theo độ tuổi  if 'Age' in data.columns:      print("Tỷ lệ hủy dịch vụ theo độ tuổi:")      print(data.groupby("Age")["Churn"].mean())  # Hiển thị một số dòng đầu tiên của dữ liệu đã được tính toán độ tuổi (nếu có)  if 'Age' in data.columns:      print(data[['Birthdate', 'Age']].head()) |

*Hình 2.4.3 Code python sử dụng phương pháp groupby để nhóm dữ liệu theo phân loại.*

Và kết quả :

|  |
| --- |
|  |

*Hình 2.4.4 Kết quả sử dụng phương pháp groupby để nhóm dữ liệu theo phân loại.*

Giải thích:

* Tỷ lệ hủy dịch vụ khá đồng đều giữa các hạng mục phí bảo hiểm, dao động từ khoảng 0.632 đến 0.640.
* Những người có BMI từ 18 đến 24 có tỷ lệ hủy dịch vụ là 1.0 (100%), trong khi những người có BMI từ 25 đến 28 có tỷ lệ hủy dịch vụ là 0.0 (0%).

**2.4.3 Phân tích trực quan:**

* Sử dụng thư viện Matplotlib để tạo các biểu đồ hình (bar chart, pie chart, histogram, box plot) nhằm trực quan hóa dữ liệu và các mối quan hệ giữa các biến số.

|  |
| --- |
| # Phân tích trực quan bằng Matplotlib  import pandas as pd  import matplotlib.pyplot as plt  data = pd.read\_csv('randomdata.csv')  # Chuyển đổi cột Churn sang kiểu số (0 và 1)  data['Churn'] = data['Churn'].apply(lambda x: 1 if x == 'Yes' else 0)  # Biểu đồ hình cột thể hiện tỷ lệ hủy dịch vụ theo hạng mục phí bảo hiểm  churn\_by\_category = data.groupby("Category Premium")["Churn"].mean()  churn\_by\_category.plot(kind="bar")  plt.xlabel("Hạng mục phí bảo hiểm")  plt.ylabel("Tỷ lệ hủy dịch vụ")  plt.title("Tỷ lệ hủy dịch vụ theo hạng mục phí bảo hiểm")  plt.show()  # Biểu đồ phân bố (histogram) của chỉ số BMI  plt.hist(data["BMI"], bins=20)  plt.xlabel("Chỉ số BMI")  plt.ylabel("Số lượng khách hàng")  plt.title("Phân bố chỉ số BMI")  plt.show()  # Biểu đồ tròn thể hiện phân bố lý do yêu cầu bồi thường (Claim Reason)  claim\_reason\_counts = data["Claim Reason"].value\_counts()  claim\_reason\_counts.plot(kind="pie", autopct='%1.1f%%')  plt.ylabel("")  plt.title("Phân bố lý do yêu cầu bồi thường")  plt.show()  # Biểu đồ hộp (box plot) của số tiền yêu cầu bồi thường theo lý do yêu cầu bồi thường  data.boxplot(column="Claim Amount", by="Claim Reason", grid=False)  plt.xlabel("Lý do yêu cầu bồi thường")  plt.ylabel("Số tiền yêu cầu bồi thường")  plt.title("Số tiền yêu cầu bồi thường theo lý do yêu cầu bồi thường")  plt.suptitle("")  plt.show() |

*Hình 2.4.5 Code python phân tích trực quan với Matplotlib.*

Và kết quả :

|  |  |
| --- | --- |
|  |  |
|  |  |

*Hình 2.4.6 Kết quả sử dụng phân tích trực quan với thư viện Matplotlib.*

- Biểu đồ hình cột (bar chart): Tạo biểu đồ hình cột để thể hiện tỷ lệ hủy dịch vụ theo hạng mục phí bảo hiểm.

- Biểu đồ phân bố (histogram): Tạo biểu đồ phân bố để thể hiện phân bố chỉ số BMI của khách hàng.

- Biểu đồ tròn (pie chart): Tạo biểu đồ tròn để thể hiện phân bố lý do yêu cầu bồi thường (Claim Reason).

- Biểu đồ hộp (box plot): Tạo biểu đồ hộp để thể hiện số tiền yêu cầu bồi thường theo lý do yêu cầu bồi thường.

Giải thích:

* Biểu đồ hình cột (bar chart):
  + Các hạng mục phí bảo hiểm được liệt kê dọc theo trục X. Trong trường hợp này, các hạng mục bao gồm các giá trị như 399, 1875, 4794, và 14390. Đây có thể là các mức phí bảo hiểm hoặc các nhóm phân loại phí bảo hiểm khác nhau.
  + Trục Y hiển thị tỷ lệ hủy dịch vụ, được tính bằng cách lấy trung bình của biến Churn trong mỗi nhóm hạng mục phí bảo hiểm. Tỷ lệ này dao động từ 0.0 đến 1.0, trong đó 1.0 tương ứng với tỷ lệ hủy dịch vụ là 100%.
  + Mỗi cột trong biểu đồ đại diện cho tỷ lệ hủy dịch vụ của một hạng mục phí bảo hiểm. Chiều cao của mỗi cột cho biết tỷ lệ hủy dịch vụ trung bình cho từng hạng mục. Trong biểu đồ này, tất cả các cột có chiều cao gần như nhau, cho thấy rằng tỷ lệ hủy dịch vụ là tương đối đồng đều giữa các hạng mục phí bảo hiểm.
  + Biểu đồ cho thấy tỷ lệ hủy dịch vụ là khá đồng đều giữa các hạng mục phí bảo hiểm. Điều này có thể gợi ý rằng không có hạng mục phí bảo hiểm nào có tỷ lệ hủy dịch vụ đặc biệt cao hoặc thấp hơn so với các hạng mục khác.
* Biểu đồ phân bố (histogram):
  + Trục X biểu diễn các giá trị của chỉ số BMI. Trong biểu đồ này, chỉ số BMI được chia thành các nhóm (bins) với các giá trị như 18, 20, 22, 24, 26, 28.
  + Trục Y biểu diễn số lượng khách hàng tương ứng với mỗi nhóm BMI. Biểu đồ cho thấy số lượng khách hàng trong mỗi nhóm BMI.
  + Mỗi cột trong biểu đồ đại diện cho số lượng khách hàng trong một nhóm BMI cụ thể. Chiều cao của mỗi cột cho biết số lượng khách hàng có chỉ số BMI nằm trong khoảng đó.
  + Biểu đồ phân bố này cho thấy số lượng khách hàng tương đối đồng đều giữa các nhóm chỉ số BMI từ 18 đến 28. Không có nhóm nào có số lượng khách hàng vượt trội so với các nhóm khác.
* Biểu đồ tròn (pie chart):
  + Medical (54.9%): Đại diện cho phần lớn yêu cầu bồi thường, chiếm 54.9% tổng số yêu cầu
  + Other (15.1%): Chiếm 15.1% tổng số yêu cầu bồi thường.
  + Phone (15.0%): Chiếm 15.0% tổng số yêu cầu bồi thường.
  + Travel (15.0%): Chiếm 15.0% tổng số yêu cầu bồi thường.
  + Yêu cầu bồi thường liên quan đến lý do y tế (Medical) chiếm tỷ lệ lớn nhất. Các lý do khác như Other, Phone và Travel có tỷ lệ tương đương nhau, mỗi lý do chiếm khoảng 15% tổng số yêu cầu bồi thường.
* Biểu đồ hộp (box plot):
  + Medical có số tiền yêu cầu bồi thường từ 1600 đến hơn 2100.
  + Other có số tiền yêu cầu bồi thường từ hơn 0 đến hơn 250.
  + Phone có số tiền yêu cầu bồi thường từ hơn 300 đến hơn 400.
  + Travel có số tiền yêu cầu bồi thường từ hơn 400 đến hơn 800.
  + Yêu cầu bồi thường tiền liên quan đến lý do y tế (Medical) chiếm tỷ lệ lớn nhất.

**CHƯƠNG 3: PHÂN TÍCH DỮ LIỆU**

**3.1 Xử lý dữ liệu.**

**3.1.1 Chuẩn bị dữ liệu.**

Dữ liệu được tham khảo và lấy từ nguồn Kaggle. Đây là một nền tảng cung cấp nhiều bộ dữ liệu phong phú và đa dạng, rất hữu ích cho việc xây dựng và kiểm thử các mô hình học máy. Việc lựa chọn bộ dữ liệu phù hợp là bước đầu tiên và quan trọng trong quá trình xử lý dữ liệu.

Nguồn dữ liệu mẫu: <https://www.kaggle.com/datasets/usmanfarid/customer-churn-dataset-for-life-insurance-industry>

File dữ liệu đã train:

<https://colab.research.google.com/drive/14rf0JXXz_DRY8OyZKhrJVRX1wdK5gwrG#scrollTo=xLo3gD2CHfja>

Đưa dữ liệu vào driver , tôi sử dụng Google Colaboratory chuẩn bị cho việc training dữ liệu.

|  |
| --- |
|  |

*Hình 3.1.1 Chuẩn bị nguồn dữ liệu khởi tạo train model.*

Khởi tạo môi trường ảo trên Google Colaboratory và kết nối đến thư mục lưu trữ các dữ liệu trên.

|  |
| --- |
|  |

*Hình 3.1.2 Kết nối tới kho dữ liệu trên driver cá nhân.*

**3.1.2 Tiền xử lý dữ liệu.**

1. Cài đặt các thư viện python.

|  |
| --- |
|  |

*Hình 3.1.3 Cài đặt các thư viện python hỗ trợ học máy.*

* Numpy là thư viện chính để làm việc với mảng (array) trong Python. Nó cung cấp các công cụ mạnh mẽ để thao tác và thực hiện các phép toán trên các mảng đa chiều.
* Scipy bổ sung cho numpy bằng cách cung cấp các hàm toán học và khoa học cao cấp hơn, bao gồm các công cụ để tối ưu hóa, tích hợp, và xử lý tín hiệu.
* Pandas là thư viện mạnh mẽ để thao tác và phân tích dữ liệu. Nó cung cấp cấu trúc dữ liệu DataFrame, rất tiện lợi cho việc xử lý dữ liệu dạng bảng.
* Matplotlib là thư viện dùng để vẽ đồ thị và trực quan hóa dữ liệu. Nó cho phép tạo ra các biểu đồ, đồ thị và các hình ảnh khác nhau để phân tích dữ liệu.

Việc cài đặt các phiên bản thư viện đồng bộ theo từng phiên bản trong quá trình thực hành sẽ giúp phần backend Flask python sau khi đã có model sẽ đồng bộ thư viện, tránh xung đột trong quá trình code.

|  |
| --- |
| data = pd.read\_csv('randomdata.csv')  num\_columns = data.shape[1]  print(f'Số cột: {num\_columns}')  Số cột: 12 |

*Hình 3.1.4 Code import dữ liệu đầu vào.*

1. Tiền xử lý dữ liệu.
2. Kiểm tra dữ liệu đầu vào:

|  |
| --- |
| print(data.info())  <class 'pandas.core.frame.DataFrame'>  RangeIndex: 200000 entries, 0 to 199999  Data columns (total 12 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----  0 Unnamed: 0 200000 non-null int64  1 Customer Name 200000 non-null object  2 Customer\_Address 200000 non-null object  3 Company Name 200000 non-null object  4 Claim Reason 200000 non-null object  5 Data confidentiality 200000 non-null object  6 Claim Amount 200000 non-null int64  7 Category Premium 200000 non-null int64  8 Premium/Amount Ratio 200000 non-null float64  9 Claim Request output 200000 non-null object  10 BMI 200000 non-null int64  11 Churn 200000 non-null object  dtypes: float64(1), int64(4), object(7)  memory usage: 18.3+ MB  None |

*Hình 3.1.5 Code hiển thị thông tin tóm tắt của dữ liệu.*

Giải thích:

* RangeIndex 200000 entries, 0 to 199999 : DataFrame này có 200,000 hàng (bản ghi), từ 0 đến 199,999.
* Data columns (total 12 columns): DataFrame có tổng cộng 12 cột.
* Column là tên các cột kèm theo đó là số lượng giá trị các cột có null hay không.
* Và bộ nhớ dử dụng là 18.3+ MB.

Nó cung cấp cái nhìn tổng quản của dữ liệu và tôi xác định tình trạng và chuẩn bị cho các bước phân tích tiếp theo.

1. Loại bỏ các trường không gây ảnh hưởng đến quá trình phân tích dữ liệu.

|  |
| --- |
| # Loại bỏ các cột không quan trọng  columns\_to\_drop = ['Unnamed: 0', 'Customer Name', 'Customer\_Address', 'Company Name']  data\_cleaned = data.drop(columns=columns\_to\_drop)  # Hiển thị các cột còn lại  print(data\_cleaned.head()) |

*Hình 3.1.6 Code loại bỏ các cột không quan trọng đến quá trình phân tích dữ liệu.*

Định nghĩa các tên cột cần loại bỏ:

* Đây là danh sách các cột mà bạn muốn loại bỏ khỏi DataFrame. Các cột này được xác định là không quan trọng cho mục đích phân tích hoặc xây dựng mô hình.
* **Unnamed: 0**: Cột này có vẻ như là một cột chỉ mục không cần thiết.
* **Customer Name**: Tên khách hàng thường không có giá trị đối với phân tích định lượng.
* **Customer Address**: Địa chỉ khách hàng cũng thường không có giá trị trong mô hình học máy.
* **Company Name**: Tên công ty có thể không ảnh hưởng đến kết quả phân tích hoặc mô hình.

Mục đích :

* Bằng cách loại bỏ các cột không quan trọng, bạn giảm kích thước của DataFrame, giúp việc xử lý và phân tích dữ liệu trở nên nhanh chóng và hiệu quả hơn.
* Loại bỏ các cột không quan trọng giúp tập trung vào những đặc trưng (features) có khả năng ảnh hưởng đến phân tích hoặc mô hình học máy, từ đó cải thiện hiệu suất và độ chính xác của mô hình.
* Loại bỏ các thông tin không cần thiết giúp giảm nhiễu trong dữ liệu, giúp mô hình học máy tập trung vào các thông tin hữu ích và có giá trị.

Kết quả:

|  |
| --- |
|  |

*Hình 3.1.7 Kết quả loại bỏ dữ liệu không quan trọng.*

1. Kiểm tra giá trị duy nhất và số lần xuất hiện trong các cột.

|  |
| --- |
| import pandas as pd  for column in data\_cleaned.columns:      print(f"Column: {column}")      print(data\_cleaned[column].value\_counts())      print() |

*Hình 3.1.8 Code kiểm tra các giá trị duy nhất trong mỗi cột.*

Giải thích :

* Kiểm tra các giá trị duy nhất và số lần xuất hiện của chúng giúp hiểu rõ hơn về phân phối của dữ liệu trong từng cột. Điều này đặc biệt quan trọng đối với các cột phân loại, nơi mà bạn có thể thấy rõ các nhóm giá trị phổ biến và hiếm gặp.
* Nếu có những giá trị bất thường hoặc lỗi dữ liệu, chúng sẽ xuất hiện rõ ràng trong kết quả của value\_counts(). Ví dụ, nếu bạn thấy có các giá trị không hợp lệ hoặc không mong muốn, bạn có thể xử lý chúng trước khi tiếp tục với các bước phân tích hoặc xây dựng mô hình.
* Việc xem các giá trị duy nhất và số lần xuất hiện giúp bạn hiểu rõ hơn về đặc điểm của dữ liệu, từ đó có thể đưa ra các quyết định xử lý dữ liệu phù hợp. Ví dụ, bạn có thể quyết định loại bỏ các giá trị hiếm gặp hoặc hợp nhất các nhóm giá trị tương tự nhau.

|  |
| --- |
|  |

*Hình 3.1.8 Kết quả kiểm số các giá trị duy nhất trong mỗi cột.*

Trong kết quả này, bạn có thể thấy rõ ràng các giá trị duy nhất trong mỗi cột và số lần xuất hiện của chúng, giúp bạn có cái nhìn tổng quan về dữ liệu của mình.

Kiểm tra null trong tập dữ liệu hiện tại:

|  |
| --- |
| nan\_info = data\_cleaned.isnull().sum()  print(nan\_info) |

*Hình 3.1.8 Code kiểm tra dữ liệu có null.*

*Kết quả:*

|  |
| --- |
|  |

*Hình 3.1.9 Kết quả kiểm tra dữ liệu có null.*

· Khi không có giá trị null hoặc NaN, bạn không cần phải lo lắng về việc xử lý dữ liệu thiếu hoặc bổ sung giá trị.

· Điều này giúp tiết kiệm thời gian và công sức, vì không cần áp dụng các kỹ thuật như thay thế giá trị thiếu bằng giá trị trung bình, giá trị phổ biến, hoặc sử dụng các thuật toán khác để ước lượng giá trị thiếu.

Nếu có dấu hiệu các cột dữ liệu null hoặc NaN thì chúng ta có thể sử dụng phương pháp loại bỏ null hoặc NaN:

|  |
| --- |
| data\_cleaned = data\_cleaned.dropna()  print(data\_cleaned.info()) |

*Hình 3.1.10 Code clean các giá trị NaN hoặc null.*

Hoặc dùng phương pháp thay thế giá trị bị thiếu bằng cách tính các giá trị trung bình:

|  |
| --- |
| data\_cleaned['Claim Amount'] = data\_cleaned['Claim Amount'].fillna(data\_cleaned['Claim Amount'].mean())  print(data\_cleaned.info()) |

*Hình 3.1.11 Code điền giá trị trung bình cho các vị trí thiếu.*

Tương tự cho cách cột khác.

Và có thể dùng mô hình máy học dự đoán cho các giá trị bị thiếu:

|  |
| --- |
| from sklearn.impute import KNNImputer  # Tạo đối tượng KNNImputer  imputer = KNNImputer(n\_neighbors=5)  # Chỉ định các cột số để dự đoán giá trị thiếu  numerical\_columns = ['Claim Amount', 'Category Premium', 'Premium/Amount Ratio', 'BMI']  data\_cleaned[numerical\_columns] = imputer.fit\_transform(data\_cleaned[numerical\_columns])  print(data\_cleaned.info()) |

*Hình 3.1.12 Code điền giá trị cho các vị trí thiếu bằng mô hình máy học.*

1. Phân chia dữ liệu.

Ở đây tôi chia dữ liệu thành 2 phần:

* **X (Các đặc trưng)**: Chứa tất cả các cột dữ liệu mà mô hình sẽ sử dụng để học và đưa ra dự đoán. Các đặc trưng có thể bao gồm các biến số, chỉ số, hoặc bất kỳ thông tin nào mô tả đối tượng hoặc tình huống mà bạn đang phân tích.
* **Y (Mục tiêu)**: Chứa giá trị mà mô hình phải dự đoán. Đây là biến phụ thuộc hoặc biến đầu ra mà bạn đang cố gắng ước lượng.

|  |
| --- |
| X = data\_cleaned.iloc[:,:-1].values  Y = data\_cleaned.iloc[:,-1].values |

*Hình 3.1.13 Code chia dữ liệu thành tập huấn luyện và tập kiểm tra.*

Việc chia dữ liệu thành tập huấn luyện và tập kiểm tra giúp đánh giá khả năng tổng quát của mô hình và đảm bảo rằng mô hình không chỉ học thuộc lòng dữ liệu huấn luyện mà còn có khả năng dự đoán chính xác trên dữ liệu mới.

1. Đánh nhãn dữ liệu.

Lý do đánh nhãn dữ liệu:

* Các mô hình học máy thường yêu cầu dữ liệu đầu vào dưới dạng số. Dữ liệu phân loại (categorical data) như tên, địa chỉ, loại sản phẩm, v.v., không thể được sử dụng trực tiếp trong hầu hết các thuật toán học máy vì chúng không có giá trị số.
* Đánh nhãn hoặc mã hóa các cột phân loại thành dạng số giúp các mô hình học máy hiểu và xử lý chúng.
* Mô hình học máy có thể hoạt động tốt hơn và nhanh hơn khi dữ liệu đã được mã hóa thành các dạng số, vì các phép toán số học và ma trận là cơ sở cho hầu hết các thuật toán học máy.
* Mã hóa dữ liệu giúp giảm thiểu lỗi và đảm bảo rằng dữ liệu được đưa vào mô hình ở định dạng mà mô hình có thể xử lý chính xác.

Vì vậy , ở đây tôi sẽ chuyển các cột 0, 1, 5 là Claim Reason , Data confidentiality, Claim Request output và cột của Y là Churk thành dạng số như sau:

Đối với tập X:

|  |
| --- |
| from sklearn.preprocessing import OneHotEncoder  from sklearn.compose import ColumnTransformer  ct = ColumnTransformer(transformers=[('encoder', OneHotEncoder(), [0,1,5])], remainder='passthrough')  X = np.array(ct.fit\_transform(X))  print(X[0]) |

*Hình 3.1.14 Code đánh nhãn dữ liệu cho tập X.*

|  |
| --- |
| [0.0 0.0 0.0 1.0 0.0 1.0 0.0 0.0 1.0 0.0 377 4794 0.0786399666249478 21] |

*Hình 3.1.15 Kết quả đánh nhãn dữ liệu của 1 phần tử X.*

Đối với tập Y:

|  |
| --- |
| from sklearn.preprocessing import LabelEncoder  le = LabelEncoder()  Y = le.fit\_transform(Y)  Y |

*Hình 3.1.16 Code đánh nhãn dữ liệu cho tập Y.*

|  |
| --- |
| array([1, 1, 1, ..., 1, 1, 1]) |

*Hình 3.1.17 Kết quả đánh nhãn dữ liệu của Y.*

1. Chia tập dữ liệu:

|  |
| --- |
| from sklearn.model\_selection import train\_test\_split  np.random.seed(42)  X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X, Y, test\_size=0.2, random\_state=42 ) |

*Hình 3.1.18 Code chia tập dữ liệu tập huấn luyện và kiểm tra.*

* Tập huấn luyện (X\_train, Y\_train) được sử dụng để huấn luyện mô hình học máy. Mô hình học máy sẽ học từ dữ liệu này để tìm ra các mẫu, mối quan hệ và các đặc điểm quan trọng của dữ liệu.
* Tập kiểm tra (X\_test, Y\_test) được sử dụng để đánh giá hiệu suất của mô hình sau khi nó đã được huấn luyện. Bằng cách sử dụng tập dữ liệu chưa từng thấy trong quá trình huấn luyện, chúng ta có thể kiểm tra xem mô hình có thể tổng quát hóa tốt như thế nào đối với dữ liệu mới, không nằm trong tập huấn luyện.
* Bằng cách chia dữ liệu thành các tập huấn luyện và kiểm tra, chúng ta có thể phát hiện và ngăn ngừa hiện tượng quá khớp, tức là khi mô hình hoạt động rất tốt trên dữ liệu huấn luyện nhưng lại kém hiệu quả trên dữ liệu mới.
* Thiết lập random\_state=42 và np.random.seed(42) giúp đảm bảo rằng việc chia tách dữ liệu là ngẫu nhiên nhưng có thể tái lập. Điều này nghĩa là mỗi lần bạn chạy mã, kết quả chia tách sẽ giống nhau, giúp bạn có thể kiểm tra và so sánh kết quả một cách nhất quán.

1. Feature scaling:

|  |
| --- |
| from sklearn.preprocessing import StandardScaler  # Assume X\_train and X\_test are already defined  columns\_to\_scale = [10, 11, 13]  # Convert data to float type  X\_train\_scaled = X\_train.astype(float)  X\_test\_scaled = X\_test.astype(float)  # Apply StandardScaler to specified columns in X\_train  scaler = StandardScaler()  X\_train\_scaled[:, columns\_to\_scale] = scaler.fit\_transform(X\_train\_scaled[:, columns\_to\_scale])  # Use the same scaler to transform X\_test  X\_test\_scaled[:, columns\_to\_scale] = scaler.transform(X\_test\_scaled[:, columns\_to\_scale])  # Print the first row before and after scaling  row\_index = 100  print("Dữ liệu của hàng số", row\_index, "trước khi chuẩn hóa:")  print(X\_test[row\_index])  print("\nDữ liệu của hàng số", row\_index, "sau khi chuẩn hóa:")  print(X\_test\_scaled[row\_index]) |

*Hình 3.1.19 Code scale miền dữ liệu.*

* Chuẩn hóa giúp đảm bảo rằng các đặc trưng (features) của dữ liệu có cùng đơn vị đo lường và phạm vi giá trị, giúp mô hình học máy hoạt động hiệu quả hơn.
* Trong trường hợp này, StandardScaler sẽ chuẩn hóa các cột 10, 11, và 13 sao cho chúng có giá trị trung bình bằng 0 và độ lệch chuẩn bằng 1.
* Các đặc trưng có đơn vị đo lường hoặc phạm vi giá trị khác nhau có thể làm mô hình học máy học không chính xác. Chuẩn hóa giúp giảm thiểu vấn đề này.
* Sử dụng cùng một scaler để chuẩn hóa cả dữ liệu huấn luyện và kiểm tra đảm bảo rằng các đặc trưng được chuẩn hóa nhất quán, giúp mô hình tổng quát hóa tốt hơn khi gặp dữ liệu mới.

So sánh sự thay đổi trước và sau của 1 item:

|  |
| --- |
|  |

*Hình 3.1.20 Kết quả sau khi scale các cột dữ liệu.*

Kèm theo việc chuẩn hóa dữ liệu , tôi sẽ lưu file chuẩn hóa scale giúp tôi dễ dàng tái sử dụng và triển khai scale cho phần backend flask pyhon của tôi.

|  |
| --- |
| #lưu file scaler lại  scaler\_path = 'scaler.pkl'  joblib.dump(scaler, scaler\_path) |

*Hình 3.1.21 Code lưu file scale dữ liệu.*

|  |
| --- |
|  |

*Hình 3.1.22 Lưu file scale trong thư mục.*

Một số mục đích chính của việc scale:

* **Tái sử dụng**: Sau khi huấn luyện mô hình và chuẩn hóa dữ liệu, bạn có thể lưu lại đối tượng scaler để sử dụng lại trong tương lai, mà không cần phải chuẩn hóa lại dữ liệu từ đầu.
* **Đảm bảo nhất quán**: Khi sử dụng cùng một đối tượng scaler để chuẩn hóa dữ liệu huấn luyện và dữ liệu thử nghiệm, bạn đảm bảo rằng cả hai tập dữ liệu đều được chuẩn hóa theo cùng một cách. Điều này giúp mô hình hoạt động chính xác hơn.
* **Tiết kiệm thời gian**: Lưu đối tượng scaler giúp tiết kiệm thời gian vì bạn không cần phải tính toán lại các thông số chuẩn hóa (như mean và standard deviation) mỗi khi bạn cần chuẩn hóa dữ liệu mới.
* **Dễ dàng triển khai**: Khi triển khai mô hình vào môi trường sản xuất, việc có sẵn đối tượng scaler đã được lưu giúp quá trình triển khai nhanh chóng và dễ dàng hơn. Bạn chỉ cần tải lại scaler đã lưu và áp dụng nó lên dữ liệu mới.

**3.2 Huấn luyện dữ liệu**

**3.2.1 Huấn luyện dữ liệu bằng các mô hình khác như SVM.**

Mô hình Support Vector Machine (SVM) mang lại nhiều lợi ích trong việc đánh giá dữ liệu, đặc biệt là trong các bài toán phân loại và hồi quy.

Lợi ích của mô hình SVM:

* Hiệu suất cao với các tập dữ liệu nhỏ và trung bình: SVM hoạt động rất hiệu quả trên các tập dữ liệu có kích thước nhỏ đến trung bình. Nó có khả năng tìm ra siêu phẳng (hyperplane) tốt nhất để phân tách các lớp trong không gian đặc trưng.
* Khả năng phân loại chính xác: SVM có thể đạt được độ chính xác cao trong việc phân loại nhờ vào việc tối đa hóa biên giữa các lớp, giảm thiểu khả năng lỗi phân loại.
* Xử lý dữ liệu phi tuyến: SVM có thể sử dụng các hàm kernel để biến đổi dữ liệu phi tuyến vào không gian đặc trưng cao hơn, nơi mà các dữ liệu có thể trở nên tuyến tính. Điều này giúp SVM xử lý tốt các bài toán phân loại phức tạp.
* Tránh overfitting: Nhờ vào việc tối ưu hóa biên phân tách, SVM có khả năng tránh overfitting tốt hơn so với nhiều mô hình khác, đặc biệt khi sử dụng các kỹ thuật như regularization.
* Khả năng tổng quát hóa tốt: SVM tập trung vào các điểm dữ liệu quan trọng nhất (các support vectors) để xây dựng mô hình. Điều này giúp mô hình có khả năng tổng quát hóa tốt hơn khi gặp các dữ liệu chưa từng thấy trước đó.
* Làm việc tốt với các loại dữ liệu khác nhau: SVM có thể xử lý các loại dữ liệu khác nhau như dữ liệu nhị phân, đa lớp và dữ liệu không cân bằng.
* Độ bền với các đặc trưng không liên quan: SVM ít bị ảnh hưởng bởi các đặc trưng không liên quan hoặc không quan trọng trong dữ liệu, giúp mô hình tập trung vào các đặc trưng quan trọng hơn.
* Khả năng điều chỉnh thông số: SVM có nhiều tham số có thể điều chỉnh, như hệ số regularization (C), loại hàm kernel (linear, polynomial, RBF, v.v.), giúp tối ưu hóa mô hình cho từng tập dữ liệu cụ thể.

Từ tập dữ liệu đã xử lý trước đó, tôi sẽ tiến hành thử nghiệm với mô hình học máy SVM (Support Vector Machine).

|  |
| --- |
| from sklearn.svm import SVC  from sklearn.metrics import accuracy\_score  from sklearn.model\_selection import cross\_val\_score  # Các kernel cần thử  # Định nghĩa mô hình SVM  svm\_model = SVC()  # Đào tạo mô hình SVM  svm\_model.fit(X\_train\_scaled, Y\_train)  # Dự đoán và đánh giá hiệu suất trên tập kiểm tra  y\_pred = svm\_model.predict(X\_test\_scaled)  accuracy = accuracy\_score(Y\_test, y\_pred)  print(f'Accuracy của SVM: {accuracy:.2f}') |

*Hình 3.1.23 Code training data với SVM.*

*Kết quả thực nghiệm:*

|  |
| --- |
| Accuracy của SVM: 1.00 |

Đánh giá kết quả:

* Nếu độ chính xác đạt mức 100%, có thể mô hình của bạn đã bị overfitting, tức là mô hình hoạt động rất tốt trên dữ liệu huấn luyện nhưng không có khả năng tổng quát hóa tốt trên dữ liệu mới hoặc dữ liệu kiểm tra. Lý dó có thể dữ liệu huấn luyện và kiểm tra không đủ đa dạng hoặc không được chia đúng cách. Hoặc mô hình quá phức tạp so với dữ liệu (quá nhiều feature hoặc kernel phức tạp).
* Nếu dữ liệu của bạn rất đơn giản và có sự phân tách rõ ràng giữa các lớp, mô hình SVM có thể đạt được độ chính xác hoàn hảo mà không gặp phải overfitting.

**3.2.2 Huấn luyện dữ liệu bằng các mô hình stacking (Super learning) .**

1. Sử dụng các thư viện:

|  |
| --- |
| from sklearn.linear\_model import LogisticRegression  from sklearn.tree import DecisionTreeClassifier  from sklearn.ensemble import RandomForestClassifier  from sklearn.metrics import accuracy\_score  from sklearn.ensemble import StackingClassifier  from sklearn.datasets import load\_iris  import joblib |

*Hình 3.1.24 Code import các thư viện cho stacking.*

1. Định nghĩa các mô hình cơ sở (Base Learners):

|  |
| --- |
| base\_learners = [      ('lr', LogisticRegression()),      ('dt', DecisionTreeClassifier()),      ('rf', RandomForestClassifier())  ] |

*Hình 3.1.25 Định nghĩa các mô hình cơ sở.*

* **Logistic Regression (lr)**: Mô hình hồi quy logistic, thích hợp cho các bài toán phân loại nhị phân và đa lớp.
* **Decision Tree (dt)**: Mô hình cây quyết định, dễ hiểu và có khả năng xử lý dữ liệu phi tuyến.
* **Random Forest (rf)**: Mô hình rừng ngẫu nhiên, bao gồm nhiều cây quyết định để cải thiện độ chính xác và kiểm soát overfitting.

1. Định nghĩa mô hình meta (Meta Learner).

|  |
| --- |
| meta\_learner = LogisticRegression() |

*Hình 3.1.26 Định nghĩa các mô hình* meta*.*

* Mô hình meta học (final\_estimator) là Logistic Regression, sẽ học cách kết hợp đầu ra của các mô hình cơ sở để đưa ra dự đoán cuối cùng.

1. Tạo mô hình Super Learner.

|  |
| --- |
| super\_learner = StackingClassifier(estimators=base\_learners, final\_estimator=meta\_learner) |

*Hình 3.1.26* Tạo mô hình Super Learner*.*

* **StackingClassifier**: Kết hợp các mô hình cơ sở (base learners) và mô hình meta (meta learner) để tạo ra một mô hình mạnh mẽ hơn. Mô hình này sử dụng đầu ra của các mô hình cơ sở làm đầu vào cho mô hình meta.

1. Đào tạo mô hình Super Learner.

|  |
| --- |
| super\_learner.fit(X\_train\_scaled, Y\_train) |

*Hình 3.1.27* Đào tạo mô hình Super Learner*.*

* Đào tạo mô hình Super Learner với dữ liệu đã chuẩn hóa và nhãn (label) tương ứng.

1. Dự đoán và đánh giá hiệu suất.

|  |
| --- |
| # Dự đoán và đánh giá hiệu suất  y\_pred = super\_learner.predict(X\_test\_scaled)  accuracy = accuracy\_score(Y\_test, y\_pred)  print(f'Accuracy của Super Learner: {accuracy:.2f}') |

*Hình 3.1.28* Dự đoán và đánh giá hiệu suất.

* **predict**: Sử dụng mô hình đã huấn luyện để dự đoán nhãn cho tập kiểm tra.
* **accuracy\_score**: Tính độ chính xác của mô hình dựa trên dự đoán và nhãn thực tế.
* In ra độ chính xác của mô hình Super Learner.

1. Lưu mô hình vào file.

|  |
| --- |
| # Lưu mô hình vào file  joblib.dump(super\_learner, 'super\_learner\_model.pkl')  print('Mô hình đã được lưu vào file super\_learner\_model.pkl') |

*Hình 3.1.29* Lưu kết quả dự đoán vào file.

* **Joblib.dump**: Lưu mô hình đã huấn luyện vào file super\_learner\_model.pkl để có thể sử dụng lại sau này.

1. Tải lại mô hình từ file và dự đoán và đánh giá hiệu suất của mô hình đã tải lại.

|  |
| --- |
| # Tải lại mô hình từ file  loaded\_model = joblib.load('super\_learner\_model.pkl')  # Dự đoán và đánh giá hiệu suất của mô hình đã tải lại  y\_pred\_loaded = loaded\_model.predict(X\_test\_scaled)  accuracy\_loaded = accuracy\_score(Y\_test, y\_pred\_loaded)  print(f'Accuracy của mô hình đã tải lại: {accuracy\_loaded:.2f}') |

*Hình 3.1.30 Load mô hình và kiểm tra lại hiệu suất của mô hình.*

* Sử dụng mô hình đã tải lại để dự đoán nhãn cho tập kiểm tra và tính độ chính xác của nó.
* In ra độ chính xác của mô hình đã tải lại để đảm bảo rằng quá trình lưu và tải mô hình không làm thay đổi hiệu suất của nó.

1. Kết quả dự đoán:

Kết quả dự đoán cho thấy mô hình Super Learner của bạn đạt được độ chính xác hoàn hảo (1.00) cả trước và sau khi lưu và tải lại mô hình. Điều này có thể là dấu hiệu của một số tình huống:

* Nếu dữ liệu của bạn có sự phân tách rõ ràng và đơn giản, mô hình Super Learner có thể dễ dàng đạt được độ chính xác hoàn hảo. Điều này có thể xảy ra khi các đặc trưng của dữ liệu rất khác biệt giữa các lớp.
* Độ chính xác cao trên tập kiểm tra có thể là dấu hiệu của overfitting nếu dữ liệu huấn luyện và kiểm tra không được chọn ngẫu nhiên hoặc không đại diện cho toàn bộ tập dữ liệu. Điều này thường xảy ra khi mô hình nhớ quá rõ các mẫu trong tập huấn luyện.
* Với các tập dữ liệu nhỏ, mô hình có thể dễ dàng đạt được độ chính xác cao, nhưng kết quả này có thể không tổng quát hóa tốt cho dữ liệu mới hoặc dữ liệu thực tế.
* Nếu dữ liệu kiểm tra rất giống với dữ liệu huấn luyện hoặc chứa các mẫu đã được mô hình học, mô hình có thể đạt độ chính xác cao. Điều này có thể xảy ra nếu việc chia dữ liệu không đảm bảo tính ngẫu nhiên và đa dạng.
* Super Learner kết hợp nhiều mô hình cơ sở và một mô hình meta, có thể khai thác được nhiều khía cạnh của dữ liệu, giúp cải thiện độ chính xác. Nếu các mô hình cơ sở và mô hình meta đều mạnh mẽ và được chọn lựa kỹ lưỡng, mô hình tổng thể có thể đạt hiệu suất cao.

Có thể kiểm tra với dữ liệu thực tế:

|  |
| --- |
| # Dự đoán và so sánh với nhãn thực tế  import numpy as np  # Dự đoán trên tập kiểm tra  y\_pred = super\_learner.predict(X\_test\_scaled)  # In ra các nhãn dự đoán và nhãn thực tế  for i in range(len(Y\_test)):      print(f'Nhãn thực tế: {Y\_test[i]}, Nhãn dự đoán: {y\_pred[i]}')  # Kiểm tra các dự đoán sai (nếu có)  incorrect\_predictions = np.where(Y\_test != y\_pred)[0]  if len(incorrect\_predictions) == 0:      print("Tất cả các dự đoán đều chính xác.")  else:      print(f'Các dự đoán sai tại các vị trí: {incorrect\_predictions}')      for idx in incorrect\_predictions:          print(f'Vị trí: {idx}, Nhãn thực tế: {Y\_test[idx]}, Nhãn dự đoán: {y\_pred[idx]}') |

*Hình 3.1.31 Kiểm tra qua các dòng dữ liệu thực tế của data.*

Kết quả:

|  |
| --- |
|  |

*Hình 3.1.32 Kết quả kiểm tra từng dòng.*

Kiểm tra các chỉ số đánh giá khác:

|  |
| --- |
| from sklearn.metrics import classification\_report, confusion\_matrix  # Báo cáo phân loại  print("Báo cáo phân loại:")  print(classification\_report(Y\_test, y\_pred))  # Ma trận nhầm lẫn  print("Ma trận nhầm lẫn:")  print(confusion\_matrix(Y\_test, y\_pred)) |

*Hình 3.1.33 Code kiểm tra các chỉ số khác.*

|  |
| --- |
|  |

*Hình 3.1.34 Kết quả kiểm tra các chỉ số khác.*

Kết quả nhận xét:

* **Precision**: Tất cả các giá trị đều là 1.00, cho thấy rằng tất cả các dự đoán dương tính của mô hình đều chính xác.
* **Recall**: Tất cả các giá trị đều là 1.00, cho thấy rằng mô hình đã tìm ra tất cả các trường hợp dương tính trong dữ liệu kiểm tra.
* **F1-score**: Tất cả các giá trị đều là 1.00, cho thấy sự cân bằng hoàn hảo giữa precision và recall.
* **Support**: Số lượng mẫu trong mỗi lớp (0: 21821, 1: 38179).
* [[21821 0], [0 38179]]: Ma trận nhầm lẫn cho thấy rằng mô hình không có bất kỳ dự đoán sai nào. Tất cả các mẫu trong lớp 0 và lớp 1 đều được dự đoán chính xác.

**CHƯƠNG 4: XÂY DỰNG VÀ TRIỂN KHAI ỨNG DỤNG DỰ BÁO KHÁCH HÀNG RỜI BỎ DỊCH VỤ BẢO HIỂM**

* 1. **Triển khai xây dựng ứng dụng dự báo rời bỏ dịch vụ bảo hiểm.**
     1. **Xây dựng ứng dụng backend.**

Với ứng dụng này, tôi sẽ triển khai ứng dụng dự đoán khách hàng trên nền tảng python. Sau đây là các bước cài đặt:

* Bước 1: Cài đặt môi trường python với version 3.x . Bạn có thể tải Python từ python.org.
* Bước 2: Tạo dự án có tên là BE -> cd BE -> python -m venv venv để tạo dự án và tạo forder lưu trữ các thư viện của python trong forder venv.
* Bước 3: Kích hoạt môi trường ảo để có thể cài đặt các thư viện:
* Trên window : venv\Scripts\activate
* Trên macOS và Linux: source venv/bin/activate
* Bước 4: Cài đặt các gói thư viện như Flask, numpy, joblib, pandas,scikit-learn,... Có thể chạy lệnh pip install -r requirements.txt với các thư viện đã cài đặt trọng dữ án của tôi.

1. Xác định input đầu vào của ứng dụng.

Để triển khai ứng dụng dự đoán khả năng khách hàng rời bỏ dịch vụ bảo hiểm, đầu tiên tôi cần xác định dữ liệu đầu vào. Dữ liệu đầu vào sẽ bao gồm các đặc trưng (features) của khách hàng như BMI, lý do khách hàng yêu cầu bồi thường, mức độ bảo mật dữ liệu, số tiền yêu cầu bồi thường, khoảng chi phí tương ứng với từng loại yêu cầu bòi thường, tỷ lệ giữa phí bảo hiểm và số tiền yêu cầu bồi thường, kết quả của yêu cầu bồi thường, có thể là "Yes" (chấp nhận) hoặc "No" (từ chối).

|  |
| --- |
| {      "Claim\_Reason": "Travel",      "Data\_confidentiality": "Low",      "Claim\_Amount": 218,      "Category\_Premium": 4794,      "Premium\_Amount\_Ratio": 0.05,      "Claim\_Request\_output": "No",      "BMI": 20  } |

*Hình 4.1.1 Dữ liệu mẫu đầu vào.*

Với các biến dữ liệu , tôi đã kiểm tra trong dữ liệu gốc, mỗi biến sẽ ràng buộc trong khoảng riêng.

* Claim Reason: "Medical", "Other", "Phone", "Travel"
* Data confidentiality: "High","Very low", "Medium", "Low"
* Claim Amount: Giá trị từ 218 đến 1440
* Category Premium: 4794, 14390, 399, 1875
* Premium/Amount Ratio: Giá trị từ 0 đến 1%
* Claim Request output: "No", "Yes"
* BMI: Chỉ số khối cơ thể, ví dụ từ 18 đến 28

Giải thích:

* Claim Reason (Medical,Other, Phone, Travel) ->Lý do khách hàng yêu cầu bồi thường (ví dụ: Du lịch, Y tế).
* Data confidentiality (High, Very low, Medium, Low) ->Mức độ bảo mật dữ liệu, có thể là "Low" (thấp), "Medium" (trung bình), hoặc "High" (cao).
* Claim Amount ( from 218 to 1440 ) ->Số tiền yêu cầu bồi thường của khách hàng.
* Category Premium (Unique values: [4794, 14390, 399, 1875]) ->Khoản phí bảo hiểm tương ứng với loại hình yêu cầu bồi thường.
* Premium/Amount Ratio (0 -> 1%) ->Tỷ lệ giữa phí bảo hiểm và số tiền yêu cầu bồi thường.
* Claim Request output ( No, Yes) ->Kết quả của yêu cầu bồi thường, có thể là "Yes" (chấp nhận) hoặc "No" (từ chối).
* BMI () ->Chỉ số khối cơ thể (Body Mass Index) của khách hàng.

1. Xây dựng code module backend.

Cấu trúc thư mục của ứng dụng flask python cơ bản như sau:

|  |
| --- |
|  |

*Hình 4.1.2 Cấu trúc thư mục của ứng dụng python.*

Các thư viện được sử dụng như sau:

|  |
| --- |
| import joblib  import numpy as np  import os  from flask import Flask, jsonify, request  from flask\_cors import CORS  from dotenv import load\_dotenv  from Lable import converToValue  app = Flask(\_\_name\_\_) |

*Hình 4.1.3 Thư viện sử dụng chính.*

Triển khai các gói middleware cần thiết để sử dụng:

|  |
| --- |
| app = Flask(\_\_name\_\_)  load\_dotenv()  app = Flask(\_\_name\_\_)  CORS(app) |

*Hình 4.1.4 Code các gói middleware.*

* Gói flask dùng để chạy backend Flask với python. Triển khai các ứng dụng api từ phía server lên phía client.
* Gói CORS dùng để cho phép 1 hoặc nhiều ứng dụng frontend có thể lấy và giao tiếp với server này.

Tải các tập dữ liệu modle và scale lên:

|  |
| --- |
| # Tải mô hình từ file  model\_path = 'super\_learner\_model.pkl'  if not os.path.isfile(model\_path):      raise FileNotFoundError(f"Mô hình không được tìm thấy tại {model\_path}")  model = joblib.load(model\_path)  print('Mô hình đã được tải từ file super\_learner\_model.pkl')  # Tải scaler từ file  scaler\_path = 'scaler.pkl'  if not os.path.isfile(scaler\_path):      raise FileNotFoundError(f'Scaler không được tìm thấy tại {scaler\_path}')  scaler = joblib.load(scaler\_path)  print('Scaler đã được tải từ file scaler.pkl') |

*Hình 4.1.5 Code load dữ liệu train và file scale.*

Tái Sử Dụng Mô Hình Đã Được Huấn Luyện:

* Mô hình máy học (super\_learner\_model.pkl) đã được huấn luyện trên một tập dữ liệu cụ thể. Việc lưu và tải lại mô hình này giúp tiết kiệm thời gian và tài nguyên vì không cần phải huấn luyện lại mô hình từ đầu mỗi khi khởi động ứng dụng.
* Khi bạn lưu mô hình sau khi huấn luyện, bạn đảm bảo rằng mô hình sẽ không thay đổi trừ khi bạn chủ động huấn luyện lại và lưu mô hình mới. Điều này giúp duy trì sự nhất quán trong các dự đoán.

Chuẩn Hóa Dữ Liệu Đầu Vào:

* Scaler (scaler.pkl) đã được huấn luyện để chuẩn hóa dữ liệu đầu vào dựa trên tập dữ liệu huấn luyện ban đầu. Việc sử dụng scaler này giúp đảm bảo rằng dữ liệu mới sẽ được chuẩn hóa theo cùng một cách như dữ liệu huấn luyện, làm tăng độ chính xác của dự đoán.
* Dữ liệu đầu vào cần được chuẩn hóa theo cùng một thang đo như dữ liệu mà mô hình đã được huấn luyện. Điều này giúp mô hình hoạt động hiệu quả và đưa ra dự đoán chính xác hơn.

Triển khai một route trong ứng dụng Flask để nhận dữ liệu từ yêu cầu POST, xử lý dữ liệu, chuẩn hóa và dự đoán khả năng khách hàng rời bỏ dịch vụ bảo hiểm bằng cách sử dụng mô hình đã được huấn luyện.

|  |
| --- |
| @app.route('/', methods=['POST'])  def home():      try:          # Nhận dữ liệu từ yêu cầu POST          data = request.get\_json()          conver\_data = [              data.get('Claim\_Reason', 'Travel'),  # 'Travel' là giá trị mặc định              data.get('Data\_confidentiality', 'Low'),  # 'Low' là giá trị mặc định              data.get('Claim\_Amount', 377),  # 377 là giá trị mặc định              data.get('Category\_Premium', 4794),  # 4794 là giá trị mặc định              data.get('Premium\_Amount\_Ratio', 0),  # 0.0786399666249478 là giá trị mặc định              data.get('Claim\_Request\_output', 'No'),  # 'No' là giá trị mặc định              data.get('BMI', 21)          ]            # Chuyển đổi dữ liệu đầu vào thành các giá trị one-hot và giá trị số          claim\_reason\_one\_hot, data\_confidentiality\_one\_hot, claim\_request\_output\_one\_hot = converToValue(              conver\_data[0], conver\_data[1], conver\_data[5]          )            # Tạo mảng kết quả kết hợp các giá trị one-hot và các giá trị số khác          result\_array = np.concatenate((              claim\_reason\_one\_hot,              data\_confidentiality\_one\_hot,              claim\_request\_output\_one\_hot,              [conver\_data[2], conver\_data[3], conver\_data[4], conver\_data[6]]          ))            # Chuyển đổi dữ liệu thành dạng số và chuẩn hóa các cột cần thiết          result\_array = np.array(result\_array).reshape(1, -1).astype(float)          columns\_to\_scale = [10, 11, 13]           # Đảm bảo rằng scaler chỉ áp dụng cho cột chính xác          if len(columns\_to\_scale) > 0 and result\_array.shape[1] > max(columns\_to\_scale):              result\_array[:, columns\_to\_scale] = scaler.transform(result\_array[:, columns\_to\_scale])            # Dự đoán bằng mô hình đã tải          prediction = model.predict(result\_array)            # Chuyển đổi dự đoán thành một giá trị đơn lẻ nếu cần          result\_list = int(prediction[0])            return jsonify({"predict": result\_list})        except ValueError as ve:          return jsonify({'error': f'ValueError: {str(ve)}'}), 400      except TypeError as te:          return jsonify({'error': f'TypeError: {str(te)}'}), 400      except Exception as e:          return jsonify({'error': f'Error: {str(e)}'}), 500 |

*Hình 4.1.6 Code router api dự đoán khả năng khách hàng rời bỏ dịch vụ bảo hiểm.*

*Nhận dữ liệu từ yêu cầu POST:*

* Nhận dữ liệu từ yêu cầu POST: data = request.get\_json().
* Gán các giá trị từ dữ liệu yêu cầu vào danh sách conver\_data với các giá trị mặc định nếu không có.

Chuyển đổi dữ liệu đầu vào:

* Claim\_reason\_one\_hot, data\_confidentiality\_one\_hot, claim\_request\_output\_one\_hot = converToValue(...): Chuyển đổi các giá trị phân loại (categorical) thành dạng one-hot vectors bằng cách sử dụng hàm converToValue.
* Kết hợp các giá trị one-hot và các giá trị số khác thành mảng result\_array.

Chuẩn hóa dữ liệu:

* Result\_array = np.array(result\_array).reshape(1, -1).astype(float): Chuyển đổi mảng kết quả thành dạng numpy array và reshape thành mảng 2 chiều.
* Result\_array[:, columns\_to\_scale] = scaler.transform(result\_array[:, columns\_to\_scale]): Chuẩn hóa các cột cần thiết trong mảng kết quả sử dụng scaler.

Dự đoán:

* Prediction = model.predict(result\_array): Sử dụng mô hình đã tải để dự đoán kết quả từ dữ liệu đầu vào đã chuẩn hóa.
* Result\_list = int(prediction[0]): Chuyển đổi kết quả dự đoán thành một giá trị đơn lẻ nếu cần.

Trả về kết quả dự đoán:

* Return jsonify({"predict": result\_list}): Trả về kết quả dự đoán dưới dạng JSON.
* Kết quả này sẽ có dạng là 1 và 0 . Với 1 là ngừng sử dụng dịch vụ, 0 là tiếp tục sử dụng dịch vụ.

Chạy ứng dụng:

|  |
| --- |
| if \_\_name\_\_ == '\_\_main\_\_':      app.run(debug=True) |

*Hình 4.1.7 Code chạy ứng dụng.*

1. Chuyển đổi các giá trị phân loại (categorical values) thành các mảng one-hot, sử dụng LabelEncoder từ thư viện sklearn.preprocessing và các hàm tiện ích.

|  |
| --- |
| from sklearn.preprocessing import LabelEncoder  import numpy as np  def index\_to\_one\_hot(index, num\_classes):      one\_hot = np.zeros(num\_classes)  # Tạo mảng toàn số 0      one\_hot[index] = 1.0  # Đặt giá trị tại chỉ số tương ứng thành 1.0      return one\_hot  # Trả về toàn bộ mảng one-hot  def converToValue(claim\_reason\_value, data\_confidentiality\_value, claim\_request\_output\_value):      # Định nghĩa các nhãn      claim\_reason\_labels = ["Travel", "Medical", "Other", "Phone"]      data\_confidentiality\_labels = ["Very low", "Low", "Medium", "High"]      claim\_request\_output\_labels = ["No", "Yes"]      # Khởi tạo LabelEncoder cho từng cột      claim\_reason\_encoder = LabelEncoder().fit(claim\_reason\_labels)      data\_confidentiality\_encoder = LabelEncoder().fit(data\_confidentiality\_labels)      claim\_request\_output\_encoder = LabelEncoder().fit(claim\_request\_output\_labels)      # Chuyển đổi các giá trị chữ thành số      claim\_reason\_encoded = claim\_reason\_encoder.transform([claim\_reason\_value])[0]      data\_confidentiality\_encoded = data\_confidentiality\_encoder.transform([data\_confidentiality\_value])[0]      claim\_request\_output\_encoded = claim\_request\_output\_encoder.transform([claim\_request\_output\_value])[0]      # Chuyển đổi thành mã hóa one-hot      claim\_reason\_one\_hot = index\_to\_one\_hot(claim\_reason\_encoded, len(claim\_reason\_labels))      data\_confidentiality\_one\_hot = index\_to\_one\_hot(data\_confidentiality\_encoded, len(data\_confidentiality\_labels))      claim\_request\_output\_one\_hot = index\_to\_one\_hot(claim\_request\_output\_encoded, len(claim\_request\_output\_labels))      return claim\_reason\_one\_hot, data\_confidentiality\_one\_hot, claim\_request\_output\_one\_hot |

*Hình 4.1.8 Code phân loại giá trị thành các mảng onehot.*

Sử dụng thư viện sklearn.preprocessing.LabelEncoder, để chuyển đổi các giá trị phân loại thành các số nguyên theo sắp xếp.

Với hàm index\_to\_one\_hot() , mục đích chuyển đổi một chỉ số số nguyên thành mảng one-hot.

Với hàm converToValue(), mục đích chuyển đổi các giá trị phân loại thành các mảng one-hot.

Ví dụ:

|  |
| --- |
| # Ví dụ sử dụng  arr = converToValue('Travel', 'Medium', 'Yes')  print(arr) |

*Hình 4.1.9 Code ví dụ phân loại giá trị thành các mảng onehot.*

Kết quả:

|  |
| --- |
| (array([0., 0., 0., 1.]), array([0., 0., 1., 0.]), array([0., 1.])) |

1. Chạy trên môi trường thực nghiệm:

Để kiểm nghiệm kết quả của api này, tôi sẽ test thực nghiệm trên postman:

|  |
| --- |
|  |

*Hình 4.1.10 Chạy thực nghiệm với postman.*

*Kết quả:*

|  |
| --- |
|  |

*Hình 4.1.11 Kết quả chạy thực nghiệm với postman.*

* + 1. **Xây dựng ứng dụng frontend.**

Các bước cài đặt môi trường code giao diện:

* Bước 1: Cài đặt gói môi trường code với gói node và npm.
* Bước 2: Cài đặt yarn (vì tốc độ cài đặt thư viện của yarn nhanh hơn nhiều lần so với npm): npm install --global yarn.
* Bước 3: Khởi tạo dự án reactjs với vite: yarn create vite.
* Bước 4: Chạy dự án : cd name project -> yarn-> yarn dev.

Với dữ liệu đầu vào như postman, thì trên frontend tôi chỉ cần trả về một object dữ liệu với các tham số được yêu cầu như bên backend.

|  |
| --- |
|  |

*Hình 4.1.12 Hình ảnh giao diện web chính.*

|  |
| --- |
|  |

*Hình 4.1.13 Hình ảnh predict khách hàng vẫn sử dụng dịch vụ bảo hiểm.*

|  |
| --- |
|  |

*Hình 4.1.14 Hình ảnh predict khách hàng rời bỏ dịch vụ bảo hiểm.*

|  |
| --- |
|  |

*Hình 4.1.15 Hình ảnh giao diện web phụ.*

**KẾT LUẬN**

1. **Tổng Quan.**

Trong quá trình phát triển mô hình dự đoán churn (khách hàng rời bỏ dịch vụ), chúng ta đã sử dụng phương pháp học máy Stacking, một kỹ thuật ensemble học tập kết hợp sức mạnh của nhiều mô hình cơ sở để cải thiện hiệu suất dự đoán. Dưới đây là các kết luận chính từ quá trình triển khai và đánh giá mô hình:

1. **Hiệu Suất Cao:**

* Độ Chính Xác (Accuracy): Mô hình Stacking cho thấy độ chính xác rất cao với giá trị accuracy là 1.00 trên tập dữ liệu kiểm tra. Điều này cho thấy mô hình có khả năng phân loại chính xác các trường hợp khách hàng có khả năng rời bỏ dịch vụ bảo hiểm.
* Precision, Recall, F1-Score: Tất cả đều đạt giá trị 1.00 cho cả hai lớp (0 và 1), cho thấy mô hình không chỉ chính xác mà còn cân bằng tốt giữa precision và recall.
* Ma Trận Nhầm Lẫn (Confusion Matrix): Không có trường hợp phân loại sai nào (số lượng dự đoán đúng cho cả hai lớp đều tuyệt đối).

1. **So Sánh Với Các Mô Hình Khác.**

* So Sánh Với SVM: Mô hình SVM cũng đạt độ chính xác 1.00. Tuy nhiên, việc sử dụng Stacking cho phép tích hợp các mô hình khác nhau, giúp mô hình trở nên linh hoạt và mạnh mẽ hơn trong việc xử lý các biến động của dữ liệu thực tế.
* Kết Hợp Nhiều Mô Hình: Sử dụng Logistic Regression, Decision Tree, và Random Forest như các mô hình cơ sở và Logistic Regression làm mô hình meta. Điều này giúp tận dụng các đặc điểm mạnh của từng mô hình.
* Giảm Overfitting: Bằng cách kết hợp nhiều mô hình, Stacking giúp giảm nguy cơ overfitting, điều này đặc biệt hữu ích khi làm việc với các tập dữ liệu phức tạp.

1. **Tính Khả Dụng Thực Tế.**

* Triển Khai Thực Tế: Mô hình Stacking đã được triển khai thành công và có thể lưu trữ cũng như tải lại mà không mất đi hiệu suất. Điều này rất quan trọng cho các ứng dụng thực tế nơi mô hình cần phải hoạt động liên tục và đáng tin cậy.
* Tích Hợp Với Hệ Thống Hiện Có: Mô hình có thể được tích hợp vào các hệ thống hiện có để cung cấp dự đoán thời gian thực về khả năng khách hàng rời bỏ dịch vụ.

1. **Kết Luận.**

Mô hình dự đoán churn với Stacking đã chứng minh hiệu suất vượt trội với độ chính xác và các chỉ số đánh giá khác đạt giá trị tối đa. Việc sử dụng kỹ thuật Stacking giúp mô hình tận dụng được ưu điểm của nhiều mô hình khác nhau, giảm thiểu rủi ro overfitting và nâng cao tính khả dụng trong thực tế. Đây là một giải pháp mạnh mẽ và linh hoạt cho vấn đề dự đoán khách hàng rời bỏ dịch vụ.

**DANH MỤC TÀI LIỆU THAM KHẢO**

**Danh mục các website**

1. Reactjs : <https://legacy.reactjs.org/docs>
2. Blog staking ensemble learning: <https://viblo.asia/p/lam-chu-stacking-ensemble-learning-Az45b0A6ZxY>
3. Git repository: <https://github.com/CodexploreRepo/data-science>
4. Flask: <https://flask.palletsprojects.com/en/3.0.x/>